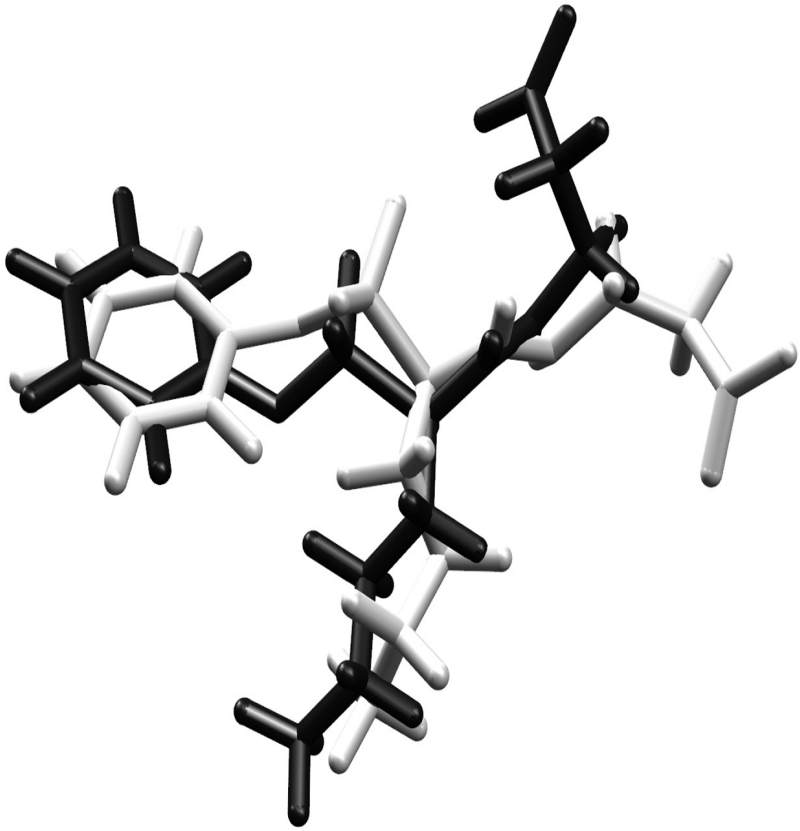
molegro virtual docker

user manual

MVD 2006.1.5 for Windows, Linux, and Mac OS X



copyright molegro 2006

Molegro ApS

Copyright © 2005–2006 Molegro ApS. All rights reserved.

Molegro Virtual Docker (MVD) and MolDock are trademarks of Molegro ApS.

All the other trademarks mentioned within this user manual are the property of their respective owners.

All trademarks are acknowledged.

Information in this document is subject to change without notice and is provided "as is" with no warranty. Molegro ApS makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Molegro ApS, shall not be liable for errors contained herein or for any direct, indirect, special, incidental, or consequential damages in connection with the use of this material.

Table of Contents

1 Introduction to Molegro Virtual Docker..... 5
1.1 Contact Information..... 6
1.2 System Requirements..... 6
1.3 Reporting Program Errors..... 6
1.4 Text Formats Used in the Manual..... 7
1.5 Screenshots Used In the Manual..... 7
1.6 Future Updates..... 7
2 Docking Tutorial..... 8
2.1 Importing and Preparing Molecules..... 8
2.2 Running the Docking Simulation..... 14
2.3 Viewing the Results..... 18
3 User Interface..... 21
3.1 Overview..... 21
3.2 Toolbar..... 22
3.3 Workspace Explorer..... 22
3.4 Properties Window..... 24
3.5 Visualization Window..... 25
3.6 Console Window..... 28
3.7 Macro and Menu Editor..... 29
3.8 Clipping Planes..... 30
3.9 Hiding Distant Residues..... 31
3.10 Workspace Finder..... 32
3.11 Sequence Viewer..... 33
3.12 Workspace Properties..... 34
3.13 Measurements and Annotations..... 34
3.14 Creating Labels..... 35
3.15 Creating Molecular Surfaces..... 37
3.16 Creating Protein Backbones..... 38
3.17 Making Screenshots..... 40
4 Preparation..... 42
4.1 Import of Molecules..... 42
4.2 Automatic Preparation..... 43
4.3 Manual Preparation..... 45
4.4 Protonation Wizard..... 47
5 Docking Functionality..... 49
5.1 Cavity Prediction..... 49
5.2 Constraints..... 50
5.3 Docking Wizard..... 53
6 Analyzing the Docking Results..... 64

6.1 Pose Organizer..... 64
6.2 Saving Molecules and Solutions Found..... 69
6.3 Pose Modifier..... 70
6.4 RMSD Matrix..... 70
6.5 Structural Alignment..... 71
7 Customizing Molegro Virtual Docker..... 72
7.1 General Preferences..... 72
7.2 Command Line Parameters 74
7.3 Changing Ranking Score Coefficients..... 75
8 Obtaining the Best Docking Results..... 76
8.1 Preparation..... 76
8.2 Docking..... 77
8.3 Post-analysis..... 78
9 Help..... 79
9.1 Online Help..... 79
9.2 Tip of the Day..... 80
9.3 The Molegro Website 80
9.4 Technical Support..... 80
10 Script Interface..... 81
10.1 Using the Script Interface..... 81
10.2 Running a Text-file Script..... 81
10.3 Examples of Common Script Jobs..... 82
10.4 Running the Script Interface Interactively..... 84
10.5 Running the Script Interface From Python..... 84
11 Appendix I: Docking Scoring Function..... 86
12 Appendix II: Docking Search Algorithm..... 90
13 Appendix III: Cavity Prediction..... 93
14 Appendix IV: Clustering Algorithm..... 94
15 Appendix V: Supported File Formats 96
16 Appendix VI: Automatic Preparation..... 98
17 Appendix VII: Third Party Copyrights..... 100
18 Appendix VIII: Keyboard Shortcuts..... 102
19 Appendix IX: Console and Macro Commands..... 103
20 Appendix X: Script Commands 110
20.1 List of Script Commands Available..... 111
20.2 Flow Control..... 121
21 Appendix XI: MolDock SE..... 124
22 Appendix XII: MolDock Score [Grid]..... 127
23 Appendix XIII: References..... 129

1 Introduction to Molegro Virtual Docker

Molegro Virtual Docker (MVD) is a program for determining the most likely conformation of how a ligand will bind to a macromolecule.

The identification of ligand binding modes is done by iteratively evaluating a number of candidate solutions (ligand conformations) and estimating the energy of their interactions with the macromolecule. The highest scoring solutions are returned for further analysis.

MVD requires a three-dimensional structure of both protein and ligand (usually derived from X-ray/NMR experiments or homology modeling). MVD performs flexible ligand docking, so the optimal geometry of the ligand will be determined during the docking.

The preferred way to get started with MVD is:

- Read the remainder of the introduction (Chapter 1)
- Go through the docking tutorial (Chapter 2).
- Read the instructions on how to use the GUI (Chapter 3).
- Read about "Obtaining the Best Docking Results" (Chapter 8).

Overall, Chapters 3 to 9 describe various aspects of MVD from importing and preparing molecules to docking and inspecting the docked solutions. Chapter 10 provides an overview of the new scripting features available from MVD v1.1.0.

More detailed information about the algorithms (cavity detection, clustering, binding mode prediction) and scoring functions (MolDock Score and MolDock Score [Grid]) used by MVD can be found in the appendices.

1.1 Contact Information

Molegro Virtual Docker is developed by:

Molegro ApS
Høegh-Guldbergs Gade 10, Building 1090
DK-8000 Aarhus C
Denmark
www.molegro.com

Information:

- Phone: (+45) 8942 3165
- Fax: (+45) 8942 3077
- VAT no.: DK 2832 6947

E-mail:

- General inquiries: info@molegro.com
- Product support: support@molegro.com
- Reporting bugs: bugs@molegro.com

1.2 System Requirements

The system requirements for Molegro Virtual Docker are:

- Windows 2000, XP, or 2003
- Linux: Most standard distributions such as Fedora Core 3 (or later versions) and Red Hat. Please send a mail to support@molegro.com if the program does not work on a particular distribution – and we will try to provide a new build.
- Mac OS X 10.3.9 (or later versions)
- Please contact us for information about running MVD on other operating systems.

1.3 Reporting Program Errors

If you discover a program error, please mail the information to:

bugs@molegro.com

Remember to specify how the error can be reproduced, the version number of Molegro Virtual Docker in question, and the operating system that was used. If possible, inclusion of molecular files used (e.g. Mol2, PDB, MVDML) will make it easier for us to reproduce (and correct) the error.

1.4 Text Formats Used in the Manual

The following formatting styles are used in this manual:

- All GUI text, labels, and keyboard shortcuts are written in bold face with initial capital letters.

  Examples: **Workspace Explorer**, **Macro Definition**, **Ctrl-O**

- Menus and menu items are identified using dividing lines and bold face.

  Example: **View** | **Docking View** indicates that the user should first select the **View** menu and then select the **Docking View** menu item.

- Filenames are written in mono-spaced font.

  Example: `\Molegro\MVD2006\bin\mvd.exe`

1.5 Screenshots Used In the Manual

The screenshots used in the manual are taken from the Windows XP version of MVD. Therefore, dialogs and other GUI related material may slightly differ on other supported OS platforms (e.g. Linux and Mac).

1.6 Future Updates

Molegro Virtual Docker contains a built-in version checker making it easy to check for new program updates including new features and bug fixes. To check for new updates, select **Help** | **Check for Updates**. A window showing available updates and details about changes made will appear (see Figure 1).

New Version Found
Latest version available
2006.1.0.3
Your current version
2006.1.0.0
Changes:
April 7th 2006: MVD 2006 Version 1.0.3
Bugfixes and some new features. See the Release Notes in the Molegro Customer Area for a full list of changes.
March 20th 2006: MVD 2006 Version 1.0.2
Bugfixes and some new features. See the Release Notes in the Molegro Customer Area for a full list of changes.
A new version has been found. Please go to www.molegro.com to update your application.
OK

Figure 1: Check for updates.

2 Docking Tutorial

This tutorial will go through a simple redocking scenario, highlighting aspects such as import and preparation of molecules, conducting the actual docking run, and visual inspection of the poses found.

2.1 Importing and Preparing Molecules

In order for MVD to be able to perform optimally, the molecules in the workspace must be properly prepared before the docking begins. The molecules can either be prepared internally in MVD, or externally by another program (e.g. MOE from CCG [CCG] or Maestro from Schrödinger, LLC [SCHRODINGER]). In this tutorial we will use the built-in preparation method available in MVD.

File Import

If the workspace is not empty, start by clearing it (select **File** | **Clear Workspace**). Next we will add some structures. This can be accomplished by selecting **File** | **Import Molecules...** or by dragging and dropping a molecule structure file. MVD supports PDB, Mol2, SDF, and its own XML-based format, MVDML.

Start by importing the file `1HVR.pdb` from the installation `examples` directory (located in the MVD installation folder). This file (a HIV-1 protease complexed with XK263) is an unmodified file taken from the RCSB Protein Data Bank (www.pdb.org).

Choosing Molecules to Import

The **Import Molecules** dialog (see Figure 2) appears.

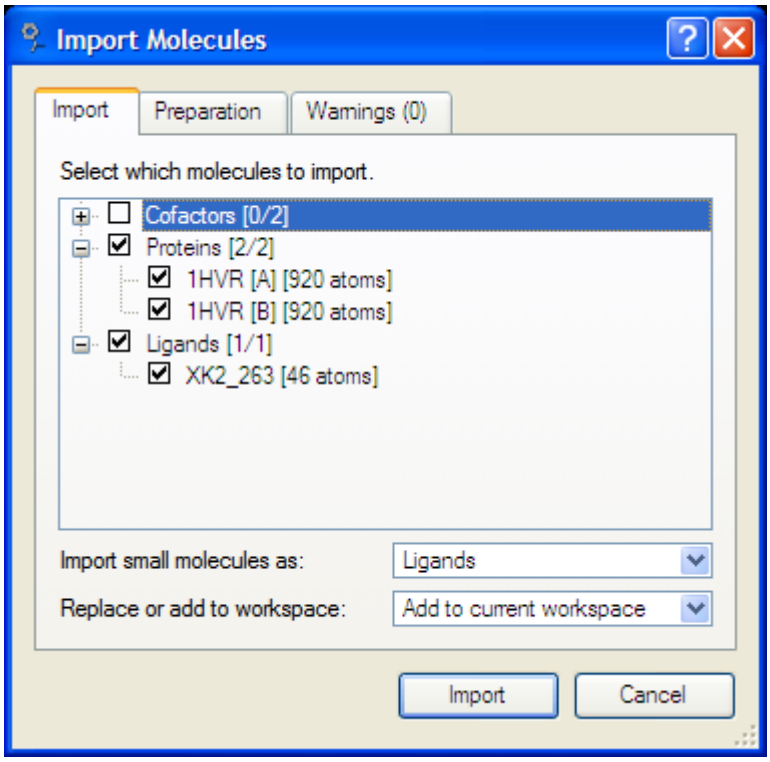


Figure 2: Importing 1HVR from the PDB file.

Deselect the cofactors since we will not need these for this example. The import dialog shows two proteins: actually these are two chains from the same protein. It also indicates that a ligand has been detected in the PDB file.

Choosing Preparation Types

Select the **Preparation** tab (see Figure 3). Some structures contain information about bond types and bond orders, and have explicit hydrogens assigned. However, PDB files often have poor or missing assignment of explicit hydrogens, and the PDB file format cannot accommodate bond order information. Set **Assign All Below** to **Always**. This ensures that all preparation will be done by MVD.

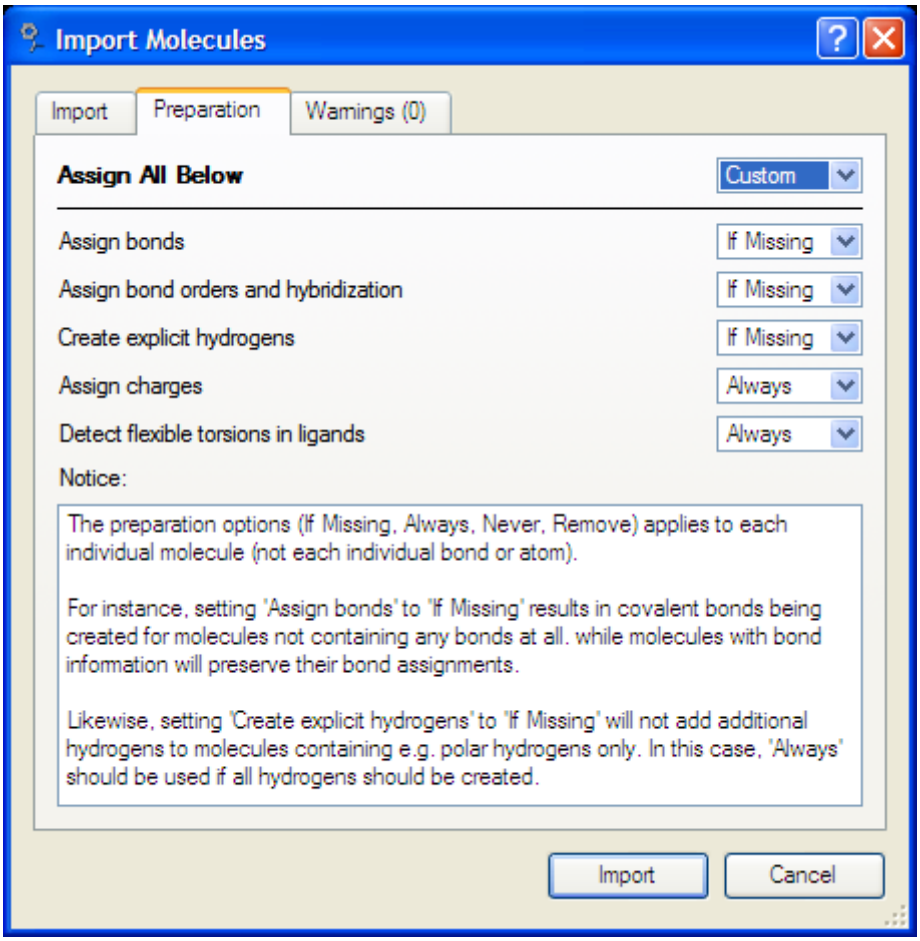

Import Molecules
Import
Preparation
Warnings (0)
Assign All Below
Custom
Assign bonds
If Missing
Assign bond orders and hybridization
If Missing
Create explicit hydrogens
If Missing
Assign charges
Always
Detect flexible torsions in ligands
Always
Notice:
The preparation options (If Missing, Always, Never, Remove) applies to each individual molecule (not each individual bond or atom).
For instance, setting 'Assign bonds' to 'If Missing' results in covalent bonds being created for molecules not containing any bonds at all. while molecules with bond information will preserve their bond assignments.
Likewise, setting 'Create explicit hydrogens' to 'If Missing' will not add additional hydrogens to molecules containing e.g. polar hydrogens only. In this case, 'Always' should be used if all hydrogens should be created.
Import
Cancel

Figure 3: Preparing the PDB-file.

Notice: If the protein structure has been prepared beforehand and saved in a format capable of handling all structural information, you should import it via the default preparation setting **If Missing**. This setting only performs a given preparation if the required information cannot be found in the file.

Inspecting the Warnings

The last tab in the import dialog (**Warnings (0)**) shows potential problems with the structure file. In this case no warnings are reported.

Now click the **Import** button. The protein and the ligand appears in the **Visualization Window** (see Figure 4).

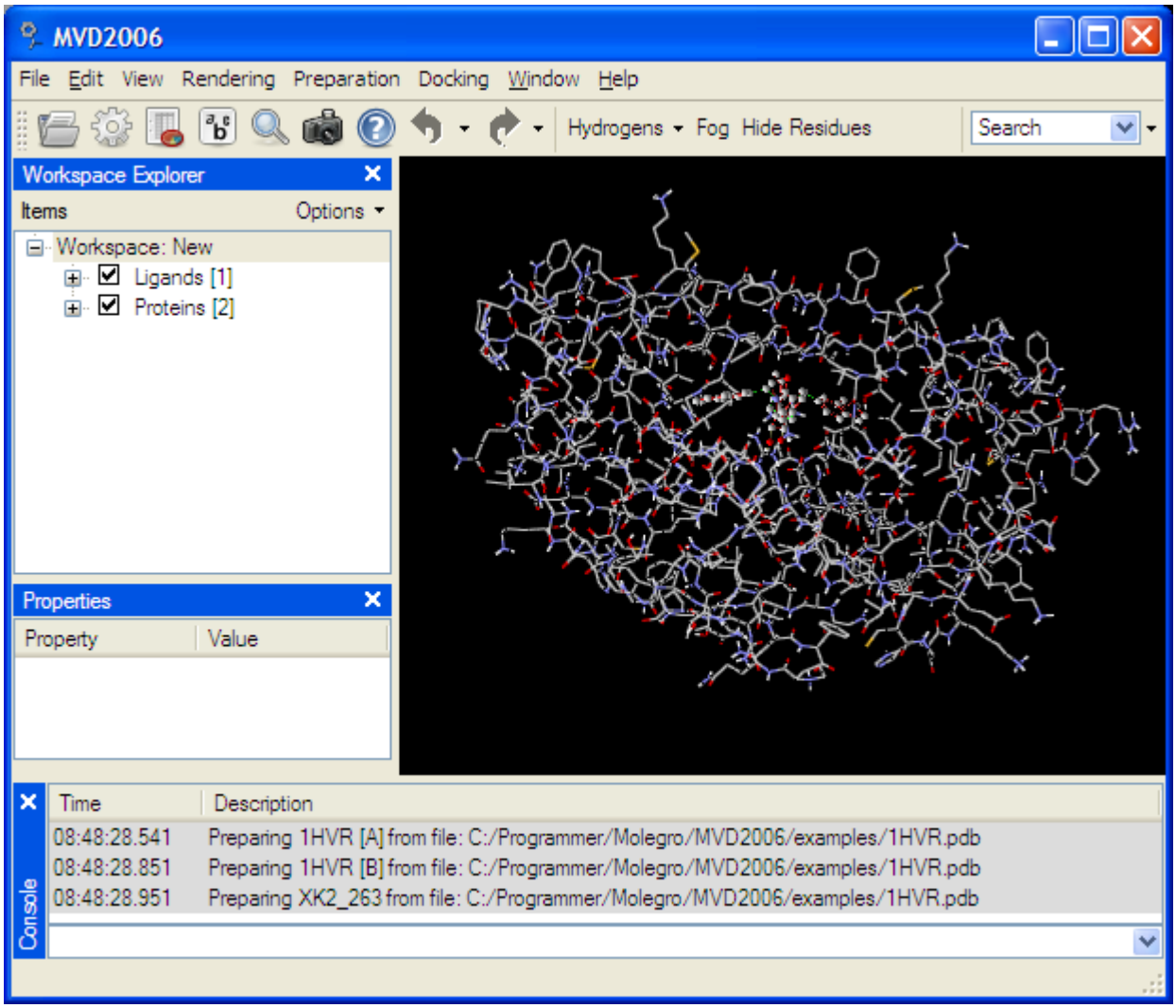


Figure 4: The imported structure.

In order to inspect the imported ligand, hide the protein by clicking on the check box next to the **Proteins** category in the **Workspace Explorer** (see Figure 5).

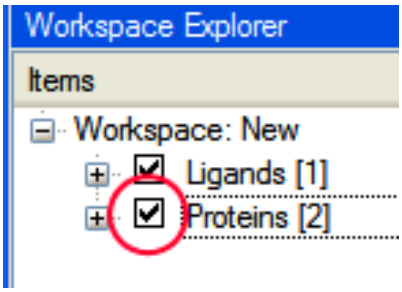


Figure 5: Hiding the Protein.

Now zoom in on the ligand (see Figure 6).

Zooming can be performed using either:

- the scroll wheel on the mouse
- by pressing and holding both mouse buttons
- by pressing shift and holding left mouse button

It is also possible to choose **Fit to Screen** from the context menu for ligands in the **Workspace Explorer**. Notice that the ligand has been assigned bond orders, aromatic rings have been detected, and explicit hydrogens have been added. Also notice that some bonds are green. These bonds will be set flexible during the docking simulation. If a bond should be held rigid during the simulation, right-click on it, and choose **Set Flexibility** from the context menu.

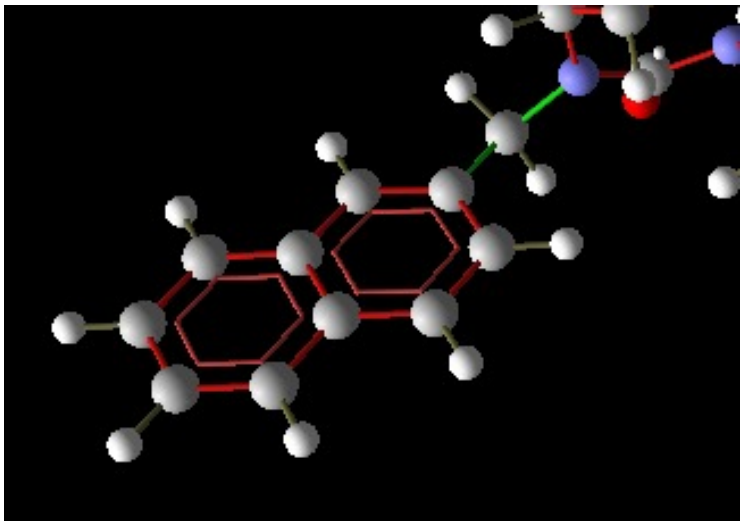


Figure 6: Inspecting flexible bonds.

Adding a Molecular Surface

Next we will add a surface to get an impression of the structure of the protein. We will do this by choosing **Create Surface...** from the **Proteins** context menu in the **Workspace Explorer** (see Figure 7).

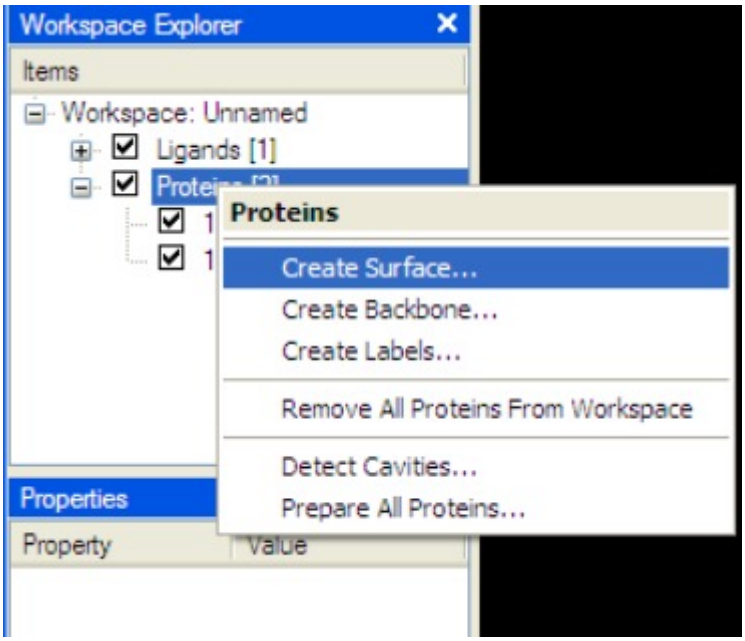


Figure 7: Surface creation.

In the dialog that appears, just click **OK**. This will create a protein surface based on the default settings which are an opaque solvent accessible surface colored according to the electrostatic potential. Notice that the surface also show up as an element in the **Workspace Explorer** (**Surfaces** category).

Predicting the Binding Site

Next we will try to narrow down the potential binding site for the protein. This can be done automatically by selecting **Preparation**| **Detect Cavities**. After pressing the **OK** button, the system will predict a binding site in the center of the protein (see Figure 8) using the algorithm described in Appendix III: Cavity Prediction.

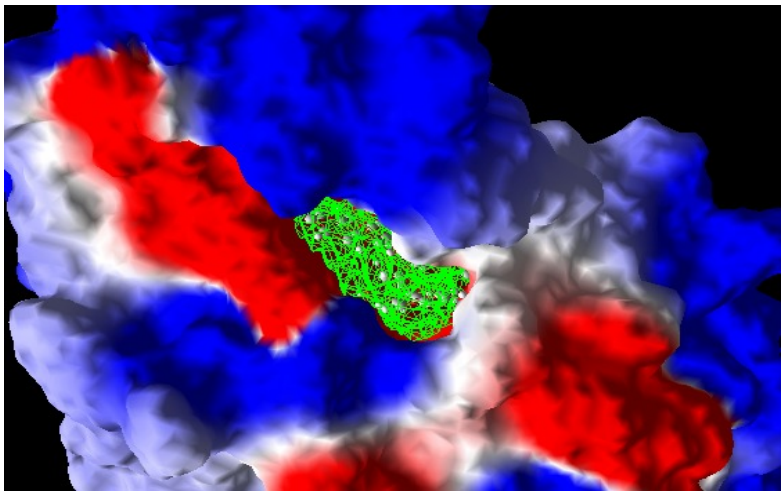


Figure 8: The predicted binding site.

Now we are ready to start the docking. To get a clearer view of this process start by selecting **View** | **Reset View!**. This will reset the 3D view (and hide the surface). Now select **View** | **Docking View**. This will switch to a view where ligands and poses have different colors and capped-stick representation instead of ball-and-stick.

2.2 Running the Docking Simulation

The **Docking Wizard** is invoked by selecting **Docking** | **Docking Wizard**.

Choosing Structures

The first tab shows which structures are included in the simulation. If multiple ligands are available they can be chosen here. Since we are doing a redocking study here, we will use the only available ligand as reference: Set **Reference Ligand** to **XK2_263** and continue to the next tab by pressing **Next**.

Defining the Region of Interest

The most important thing on the next tab is to set the binding site. Since we have detected cavities, we set **Origin** to **Cavity 1...**. If the protein had multiple potential binding sites, more choices would appear (see Figure 9).

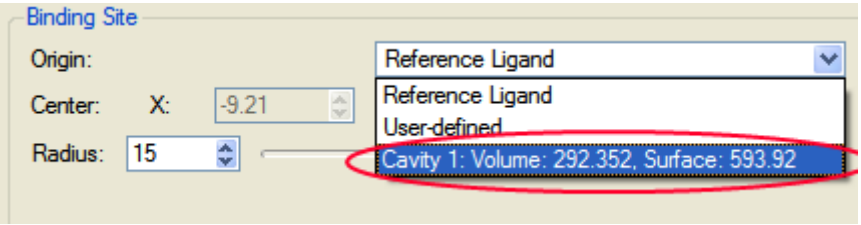


Figure 9: Selecting the binding site.

Now continue to the next tab where **Search Parameters** can be set. We will not change any search parameters – press the **Next** button to proceed to the next tab.

The next tab (**Pose Clustering**) allows you to configure whether multiple poses should be returned. We will stick to the default setting, which will limit the number of poses returned to a maximum of five. Continue to the next tab.

The **Errors and Warnings** tab in the **Docking Wizard** shows potential problems with the docking setup (if any). It should not show any warnings at this stage. Press the **Next** button to proceed to the last tab.

In the **Setup Docking Execution** tab (see Figure 10), several choices are available for executing the docking simulation. We will use the default settings (the settings are further explained in Section 5.3). Finally, the **Output directory** specifies where the docking data (log file and found poses) will be

stored. Choose a directory pressing the "**...**" button or keep the default settings.

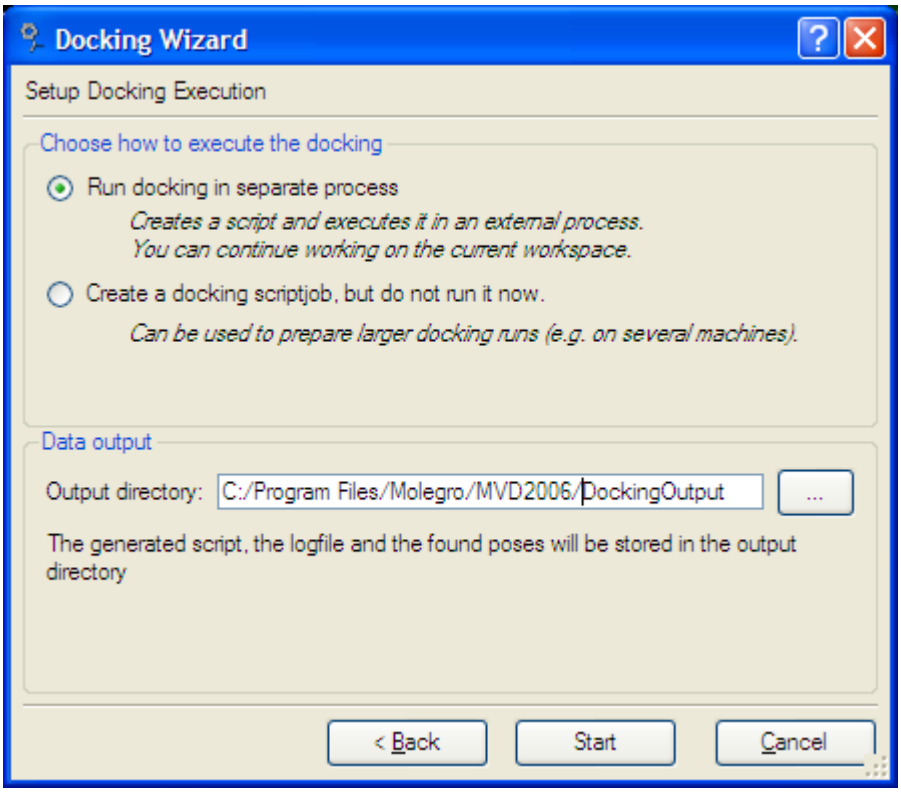

Docking Wizard
Setup Docking Execution
Choose how to execute the docking
Run docking in separate process
Creates a script and executes it in an external process.
You can continue working on the current workspace.
Create a docking scriptjob, but do not run it now.
Can be used to prepare larger docking runs (e.g. on several machines).
Data output
Output directory:
C:/Program Files/Molegro/MVD2006/DockingOutput
...
The generated script, the logfile and the found poses will be stored in the output directory
< Back
Start
Cancel

Figure 10: Setup docking execution.

Now we can begin the docking simulation by pressing the **Start** button.

The **Molegro Virtual Docker Batchjob** dialog appears showing the docking progress (see Figure 11).

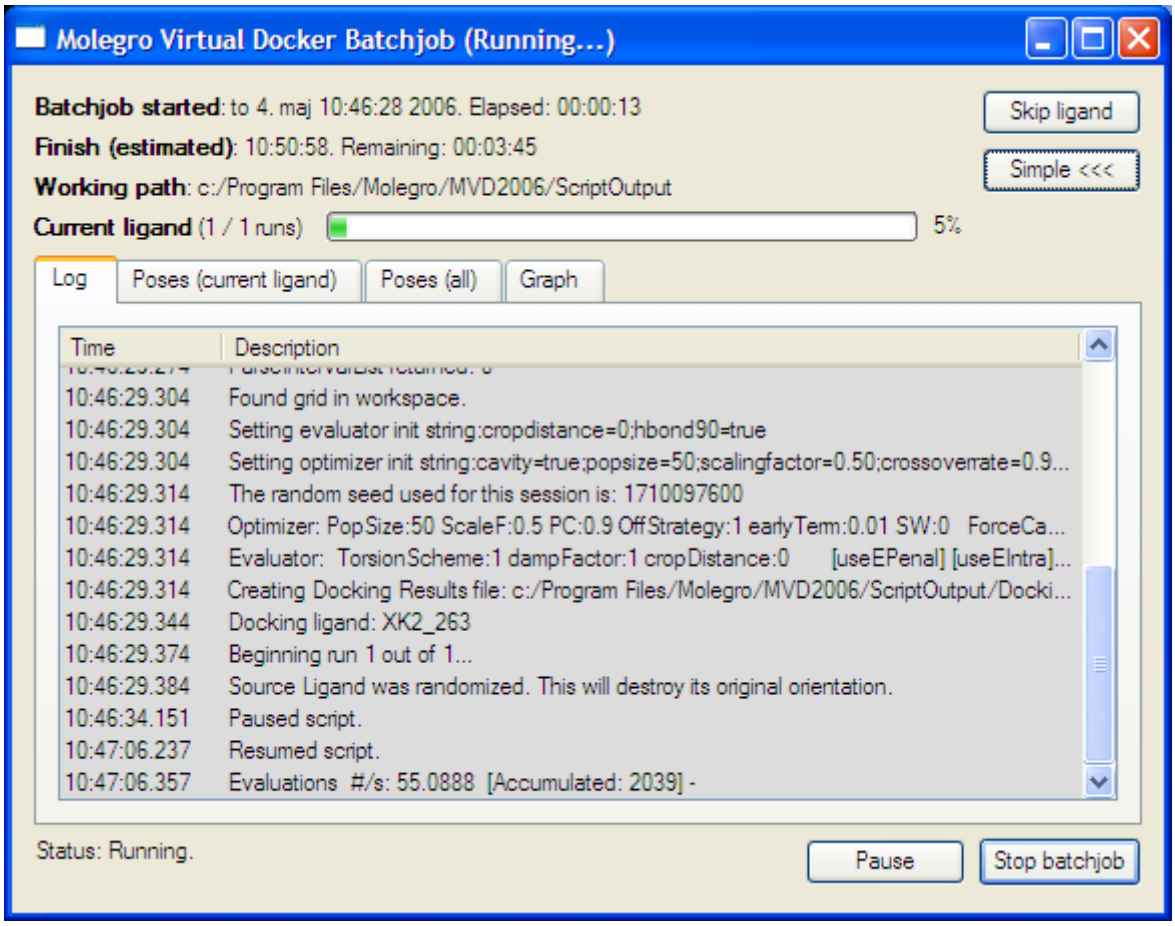

Molegro Virtual Docker Batchjob (Running...)
Batchjob started: to 4. maj 10:46:28 2006. Elapsed: 00:00:13
Finish (estimated): 10:50:58. Remaining: 00:03:45
Working path: c:/Program Files/Molegro/MVD2006/ScriptOutput
Current ligand (1 / 1 runs)
5%
Skip ligand
Simple <<<
Log
Poses (current ligand)
Poses (all)
Graph
Time
Description
10:46:29.304 Found grid in workspace.
10:46:29.304 Setting evaluator init string:cropdistance=0;hbond90=true
10:46:29.304 Setting optimizer init string:cavity=true;popsize=50;scalingfactor=0.50;crossoverrate=0.9...
10:46:29.314 The random seed used for this session is: 1710097600
10:46:29.314 Optimizer: PopSize:50 ScaleF:0.5 PC:0.9 OffStrategy:1 earlyTerm:0.01 SW:0 ForceCa...
10:46:29.314 Evaluator: TorsionScheme:1 dampFactor:1 cropDistance:0 [useEPenal] [useEIntra]...
10:46:29.314 Creating Docking Results file: c:/Program Files/Molegro/MVD2006/ScriptOutput/Docki...
10:46:29.344 Docking ligand: XK2_263
10:46:29.374 Beginning run 1 out of 1...
10:46:29.384 Source Ligand was randomized. This will destroy its original orientation.
10:46:34.151 Paused script.
10:47:06.237 Resumed script.
10:47:06.357 Evaluations #/s: 55.0888 [Accumulated: 2039] -
Status: Running.
Pause
Stop batchjob

Figure 11: Docking Progress dialog.

While the simulation is running the energy of the currently best found pose (the pose with the lowest energy) can be observed on the **Graph** tab page (see Figure 12).

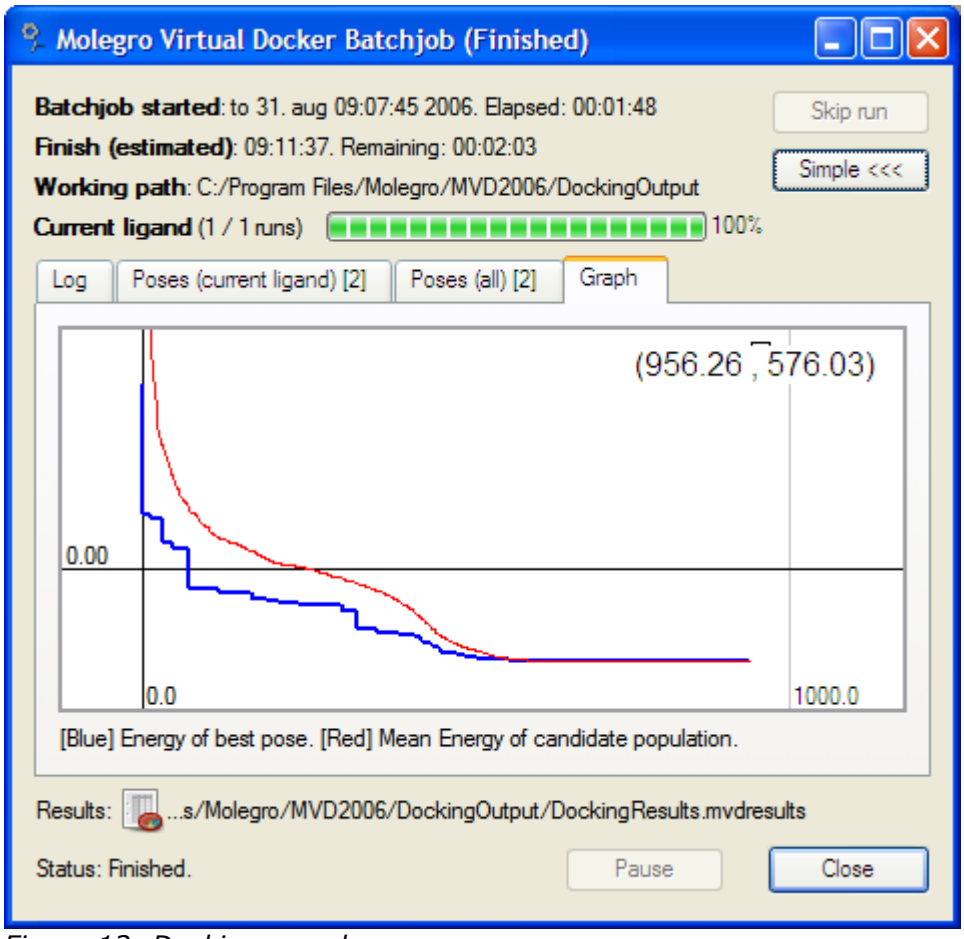


Figure 12: Docking search progress.

Let the simulation run for a while (1-2 minutes or so). The docking engine should find a good solution within 800 iterations. The simulation will eventually time out on its own (after 2,000 iterations or about 100,000 evaluations) or if the simulation has converged.

Notice: Because of the stochastic nature of the docking engine more than one docking run may be needed to identify the correct binding mode.

The docking run can also be stopped by pressing the **Stop batchjob** button. When the docking run finishes, the poses found are saved to the **Output directory** specified previously in the **Docking Wizard** dialog (here *c:\Program Files\Molegro\MVD2006\ScriptOutput* was used).

The poses found can now be imported into MVD by:

1) Selecting **Import Docking Results (*.mvdresults)...** from the **File** menu using the *DockingResults.mvdresults* file.
2) Dragging and dropping the *DockingResults.mvdresults* file onto the MVD application.
3) Dragging and dropping the DockingResults icon onto the MVD application.

The *DockingResults.mvdresults* file is located in the **Output directory** together with a docking log file and the poses found (in SDF file format).

After importing the *DockingResults.mvdresults* file, the **Pose Organizer** will appear showing the poses found (see Figure 13).

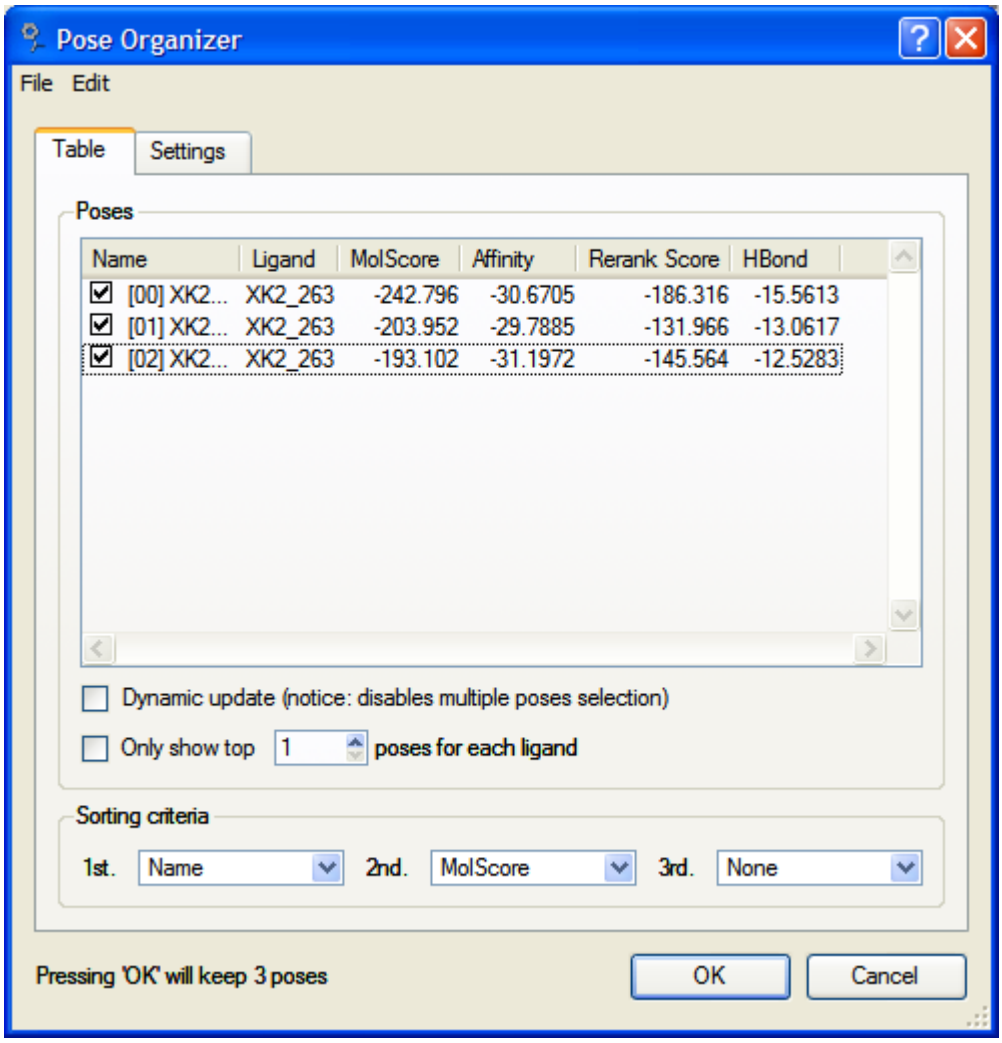

Pose Organizer
File Edit
Table Settings
Poses
Name Ligand MolScore Affinity Rerank Score HBond
[00] XK2... XK2_263 -242.796 -30.6705 -186.316 -15.5613
[01] XK2... XK2_263 -203.952 -29.7885 -131.966 -13.0617
[02] XK2... XK2_263 -193.102 -31.1972 -145.564 -12.5283
Dynamic update (notice: disables multiple poses selection)
Only show top 1 poses for each ligand
Sorting criteria
1st. Name 2nd. MolScore 3rd. None
Pressing 'OK' will keep 3 poses
OK
Cancel

Figure 13: The predicted poses.

The **Pose Organizer** allows you to inspect the poses and select which structures to keep (by toggling the select box next to them). At this point we will just add all found poses to the workspace. First select all poses by manually checking them or use the **Edit** | **Check All** menu. Afterwards, press the **OK** button.

2.3 Viewing the Results

At this point it would be a good idea to save the workspace with the new poses added. This can be done by selecting **File** | **Save Workspace...**. This will save the workspace (proteins, ligands, poses, etc.) in MVD's own XML-based

format. In order to export the poses to other formats, the **Pose Organizer** can be used.

Revisiting the Pose Organizer

First switch to the pose organizer view (**View** | **Pose Organizer View**). Each pose is shown in different colors. Next open the **Pose Organizer** (select **Docking** | **Pose Organizer**).

Turn off the original ligand and the search space sphere (colored green) in the 3D view window (by clicking the **Ligands** and **Constraints** check boxes in the **Workspace Explorer**). The poses in the **Pose Organizer** can be visualized by selecting them (see Figure 14).

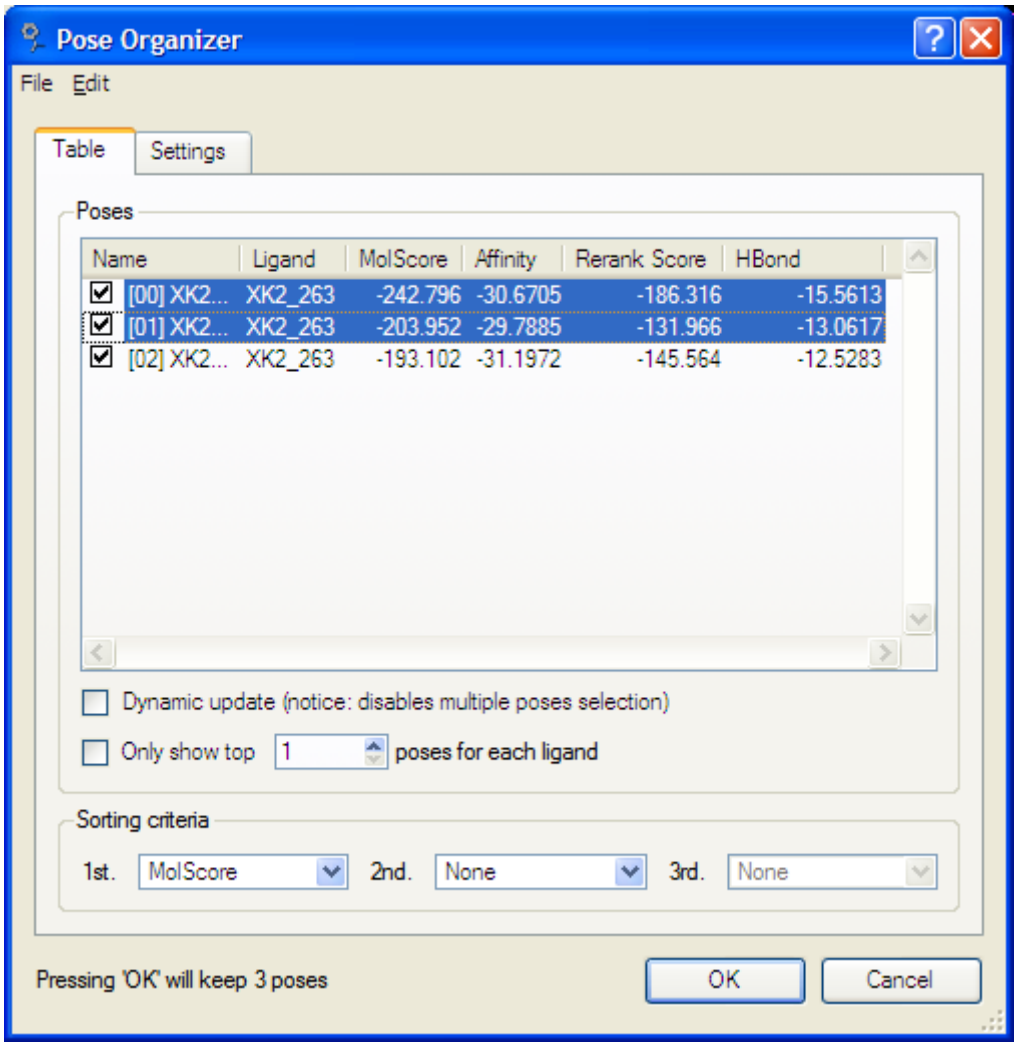


Figure 14: Viewing multiple poses.

By enabling the **Dynamic update** option, we can inspect the individual poses one at a time (single pose view mode). Click on the poses on the list to have them visualized (see Figure 15). Notice that hydrogen bonds are dynamically updated and shown when switching to a new pose.

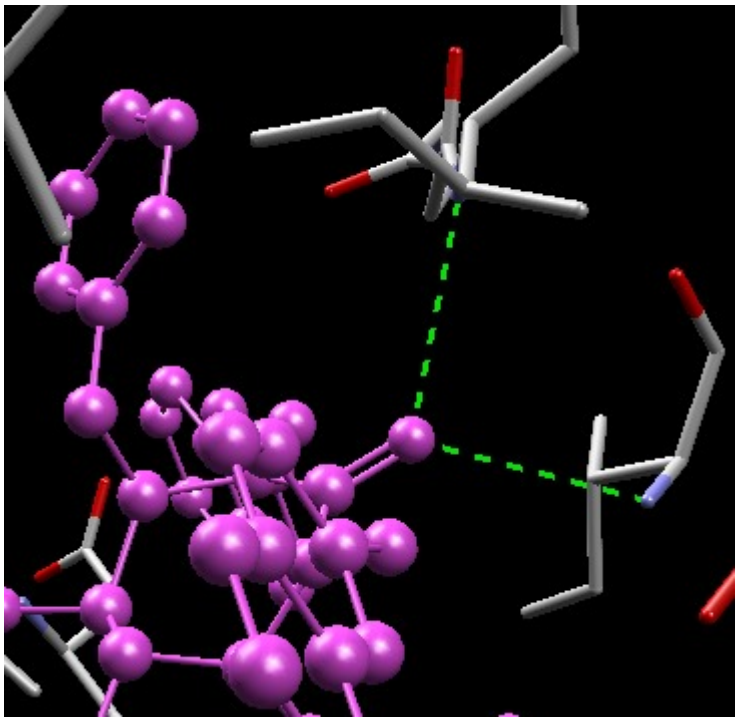


Figure 15: Viewing hydrogen bonds.

The **Pose Organizer** can also automatically rotate rotatable hydrogens (like hydroxyl rotors) in both the receptor and the ligand to their optimal position. It can also be used to rerank the ligands (using a rank score), or view their energy contributions split up into different categories (see Section 6.1 for more details).

This concludes the tutorial.

3 User Interface

3.1 Overview

The user interface in MVD is composed of a central 3D view (referred to as the **Visualization Window** or 3D world, together with a number of dockable windows (introduced below).

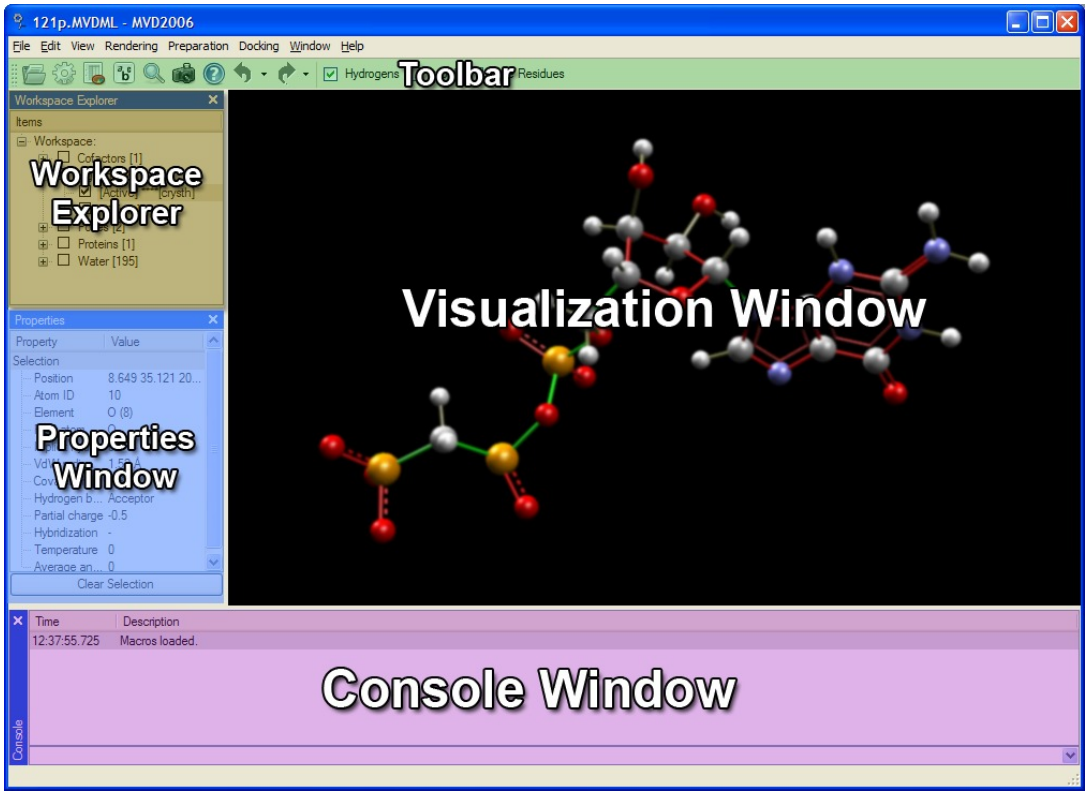


Figure 16: Main application window.

3.2 Toolbar

The **MVD Toolbar** provides easy and fast access to commonly used actions, such as import of molecules, docking using the **Docking Wizard**, and pose inspection using the **Pose Organizer**.

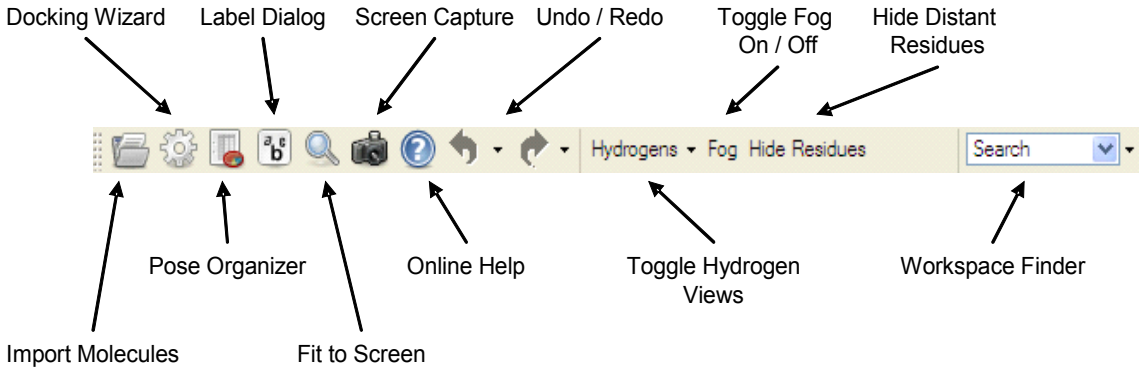


Figure 17: MVD Toolbar.

The **MVD Toolbar** also contains three toggle buttons. The **Hydrogens** button makes it easy to switch between different view modes (**Show all hydrogens**, **Show only polar hydrogens**, and **Hide all hydrogens**). The **Fog** button is used to toggle fog effects on and off. The **Hide Residues** button is used to toggle whether residues should be hidden or not (see Section 3.9 for more details). The **Workspace Finder** located at the far right side of the toolbar can be used to quickly search for molecule names and residue/atom IDs (see Section 3.10 for more details).

3.3 Workspace Explorer

The **Workspace Explorer** window (see Figure 18) contains information about the 3D-objects (both molecules, such as proteins, ligands, and structural waters - but also objects, such as labels, surfaces, backbones, and cavities).

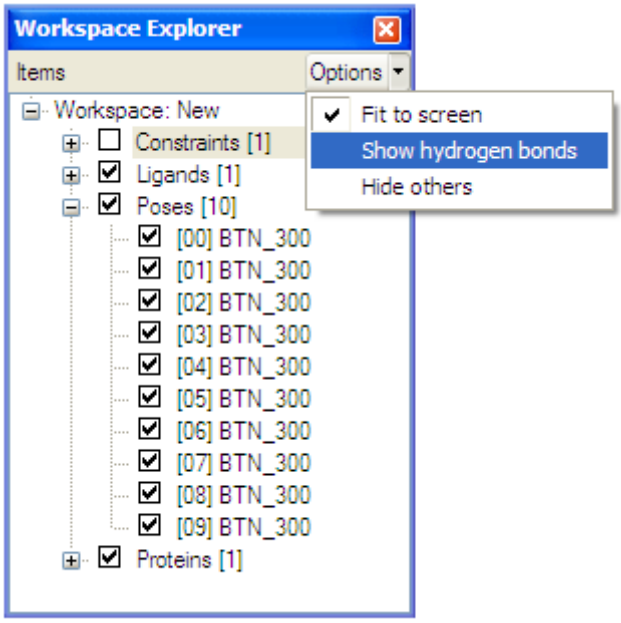


Figure 18: Workspace Explorer window.

The context menu (right mouse button click) allows the user to:

- Export molecules to PDB, Mol2, or SDF format
- Edit workspace properties
- Rename molecule
- Remove items from the current workspace
- Set the currently active ligand or reference ligand (optional)
- Copy ligands to poses (used to inspect ligands with the **Pose Organizer**)
- Clone ligand (makes a copy of the ligand)
- Convert ligand to pose or cofactor
- Convert protein to ligand
- Convert pose to ligand (used when docking poses)
- Modify ligand or pose (using the **Pose Modifier**)
- Detect cavities (using the **Cavity Prediction** dialog)
- Inspect poses (using the **Pose Organizer**)
- Prepare molecules
- Create labels, surfaces, and backbones
- Fit molecule to the visualization window

Inspecting Molecules

The **Workspace Explorer** can also be used to inspect molecules in the **Visualization Window** using the left mouse button to select the molecules or keyboard shortcuts (see below).

The **Options** button (see Figure 18) contains settings used to customize the behavior when inspecting molecules. The **Fit to screen** option will automatically zoom selected molecules so that they fit into the **Visualization Window**. The **Show hydrogen bonds** option can be used to display hydrogen bonds (only applicable for ligands and poses). The **Hide others** option toggles whether other checked molecules in the current workspace category are allowed or not.

Keyboard shortcuts are also available for inspecting molecules. Pressing the **Shift** button while clicking the left-mouse button on a molecule in the chosen category (e.g. Ligands or Poses) will fit the selected molecule in the **Visualization Window** and all other molecules located in the same category are hidden.

Alternatively, using **Ctrl+Shift** when clicking on a molecule, hydrogen bonds are shown for the selected molecule, and non-relevant protein residues are hidden.

Instead of using the mouse to select molecules to inspect, **Up** or **Down** keys can be used to browse the molecules present in the currently selected **Workspace Explorer** category. If the **Ctrl** and **Shift** shortcuts are omitted, the settings enabled in the **Options** panel will be used.

3.4 Properties Window

The **Properties Window** contains information about the currently selected (or highlighted) 3D object(s) in the **Visualization Window** and provides useful information while preparing and modifying the molecules.

Figure 19 shows an example of different properties for a highlighted atom.

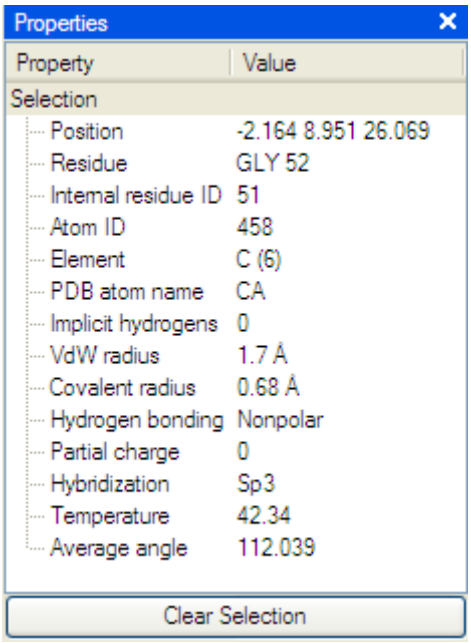


Figure 19: Example of properties for a selected atom.

3.5 Visualization Window

The **Visualization Window** (see Figure 20) visualizes all the selected molecules in the workspace and various objects to highlight import aspects (e.g. labels, annotations, charges, protonation guides, backbones, and surfaces). Moreover, cavities are visualized to allow for easy inspection of potential binding sites.

Notice: For large molecules it can be computationally slow to display all atoms. Therefore it is recommended to adjust the view to the user's needs. Often it is a good idea to add a molecular surface (perhaps transparent) to give some idea of the 3D structure. Alternatively, switching to wireframe visualization style and hiding non-polar (or all) hydrogens atoms can also improve the visualization speed significantly.

Changing the 3D World Appearance

The visualization engine is highly configurable.

Molecules can be drawn as lines (wireframe), ball-and-sticks, capped-sticks, and space-fill (CPK).

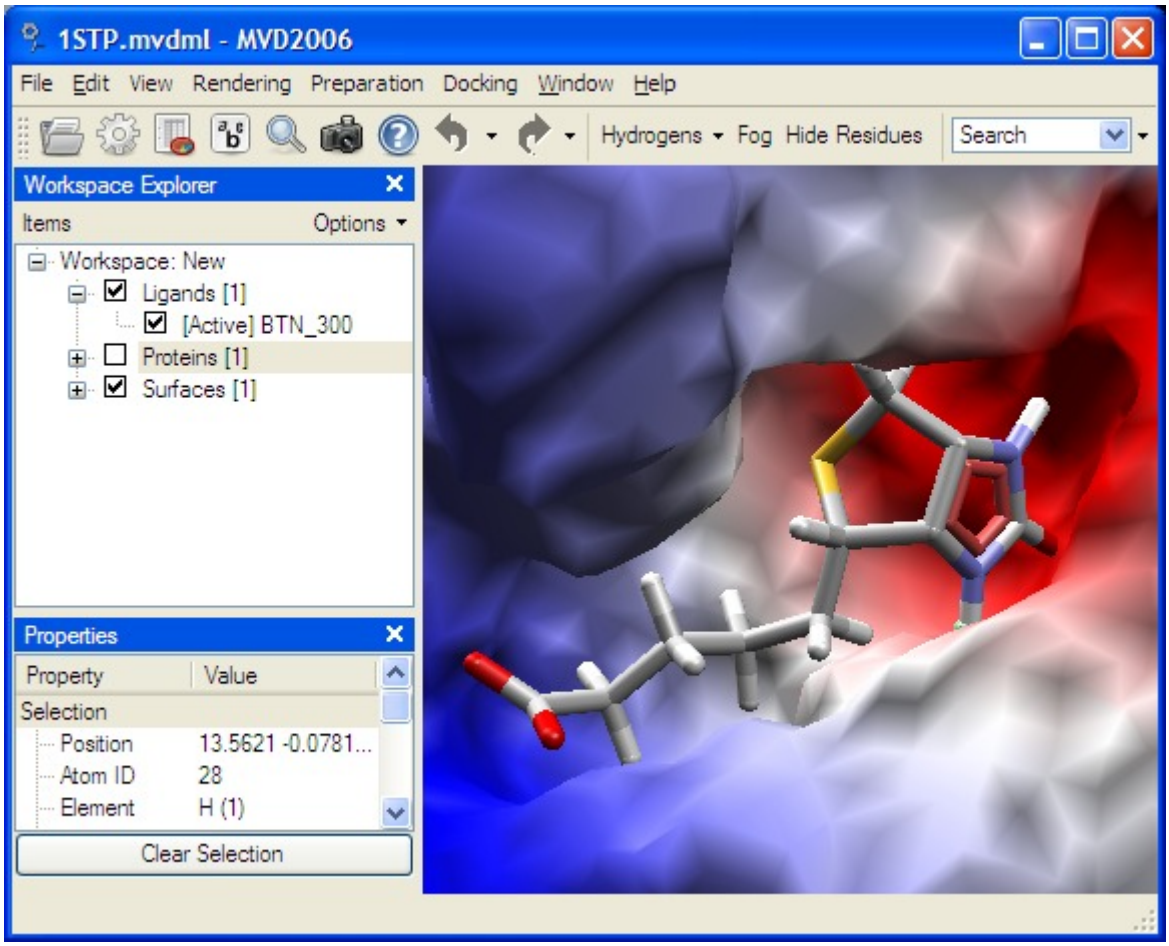


Figure 20: Visualization of Biotin (1STP) in capped-stick style and electrostatic protein surface.

Notice: Ball-and-stick is the preferred style for handling preparation of ligands, since the visualized bond shows bond order, and is color coded to display whether the bond is set rigid (brown or red) or flexible (green).

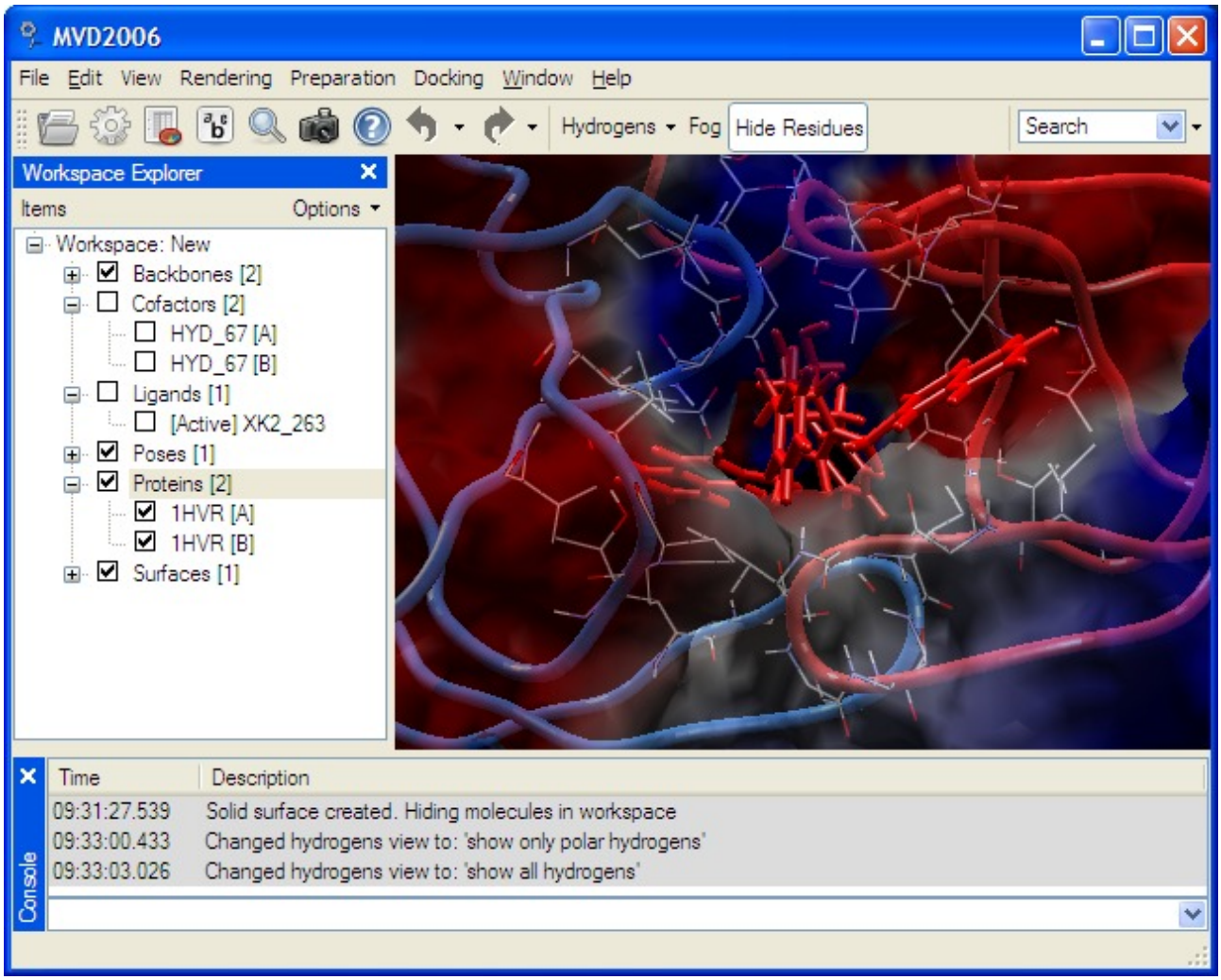


Figure 21: Main window showing different visualization styles.

The easiest way to get acquainted with the different drawing modes is to try the preset modes listed in the **Rendering** menu. Afterwards, use the **Macro and Menu Editor** (described in Section 3.7) to explore which console commands that are used for a particular view.

Navigating the 3D World

Mouse actions available in the 3D world:

Function	Action
Zoom	By pressing both mouse buttons and moving up and down. By using scroll wheel. By using shift and left mouse button.
Free Rotation	Dragging mouse cursor while holding left mouse button down.
Drag Atom Rotation	While holding mouse over an atom: Dragging mouse (left mouse button down) will force the atom to follow the mouse cursor.
Free Translation	Dragging mouse cursor while holding right mouse button down.
Show Context Menu	Click and release right mouse button.

All rotations are centered about the rotational center.

This center can be chosen by invoking the context menu on an atom (right mouse button click) and selecting **Set as Rotational Center**. Another option is to choose **Fit to Screen** from the **Workspace Explorer** context menu. **Fit to Screen** will set the rotational center to the center of the bounding box enclosing the chosen molecule. If **Fit to Screen** is invoked from the **MVD Toolbar** or from the **Visualization Window** context menu, the new rotational center will be the center of the bounding box enclosing all visible molecules in the **Visualization Window**.

Manipulating Visualization Objects

All objects in the 3D world have context menu actions. These can be used for changing their properties, e.g. setting hybridization, partial charge, implicit hydrogens, or hydrogen bond types for atoms and bond order or bond flexibility for bonds. See Section 4.3 for more details.

3.6 Console Window

The **Console Window** (at the bottom of the screen) displays information, warnings and errors. The input field at the bottom of the console window

allows the user to enter console commands. The amount of information in the console can be controlled with the associated context menu (right mouse button click) - e.g. info, warnings, and debug messages can be turned off.

3.7 Macro and Menu Editor

The **Macro and Menu Editor** allows the user to modify existing menu entries or to extend the functionality by adding new menu entries.

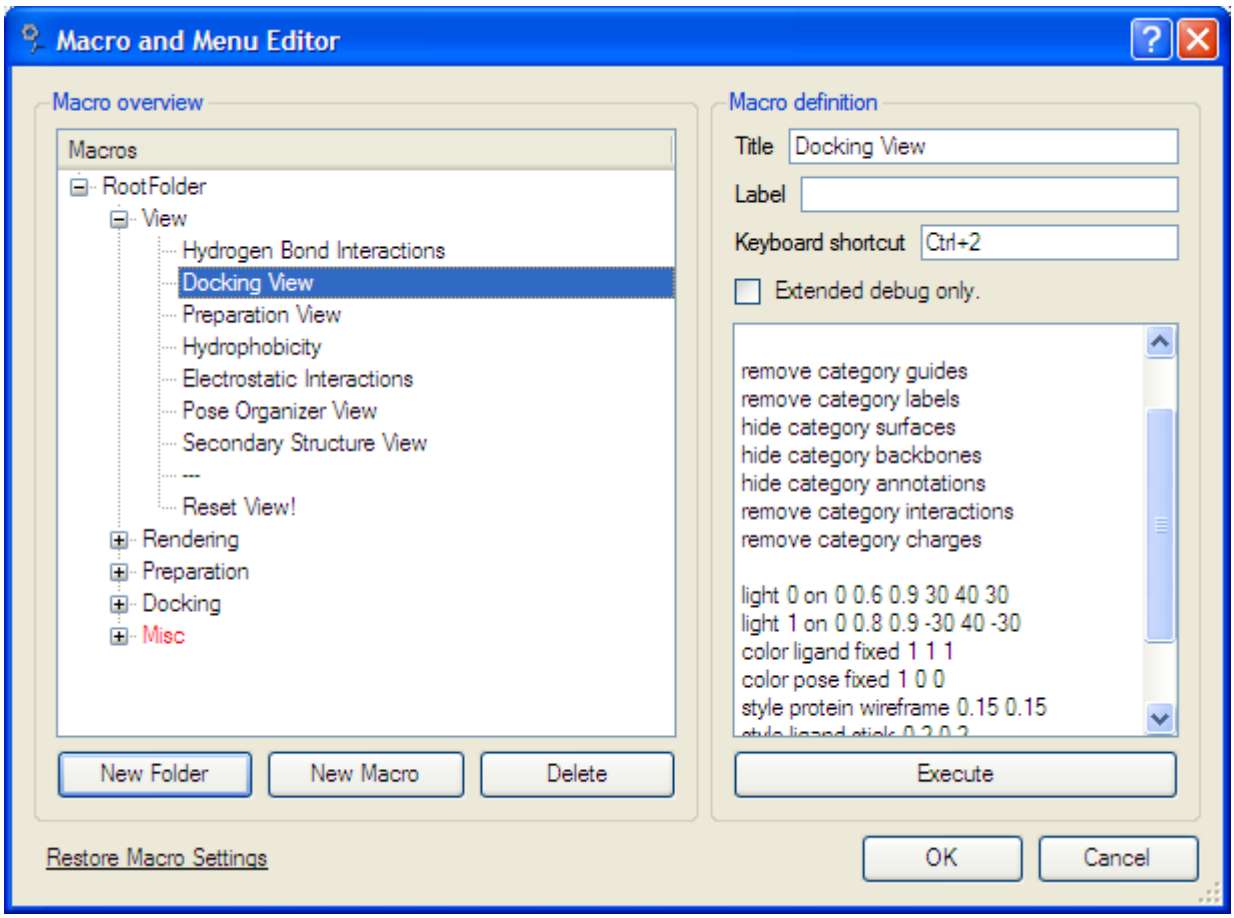


Figure 22: The Macro and Menu Editor.

It can be invoked by choosing **Edit** | **Macro and Menu Editor...** or **Window** | **Macro and Menu Editor**.

The left pane (**Macro overview**) displays a hierarchical view of all macros. The top level *folders* are mapped directly to corresponding menus in MVD. That is, **View**, **Rendering**, **Preparation** and **Docking** will appear as menus in the GUI. It is possible to add new top-level folder, by selecting the root node (**RootFolder**) and pressing the **New Folder** button.

When a folder is highlighted in the **Macro overview**, new macros can be added to it, by pressing the **New Macro** button.

New or existing macros can be modified in the right pane (**Macro definition**).

A macro consists of *a* **Title**, which is the name that is shown in the corresponding menu, an optional **Label** which can be used to assign an unique name to the macro, so that it can be called from other macros (this is done by using the macro invoke command, i.e. '!macroname'), an optional **Keyboard shortcut** (which is specified as text i.e. 'Alt+F1' or 'Ctrl+Shift+1,Shift+A' where the last shortcut simultaneously maps two alternative keyboard shortcuts) and the actual **Macro definition**.

If macros or folders appear in red in the **Macro overview**, it is because **Extended debug** is enabled for them. These items won't show up in the menus unless **Extended GUI debug properties** is enabled in the **Preferences**. This can be useful for defining macros which will not show up in the GUI, but still can be called from the **Console Window**.

It is also possible to add separators between the macros which will appear as menu separators in the GUI. To add a separator between macros just use `---` (3 strokes) as the **Title** of the macro. Similarly, separators can be created between macro folders. Again just use `---` as the **Title** of the macro folder.

Currently, rearranging the macros (e.g. changing the order of occurrence within a macro folder or moving macros between folders is not supported in the **Macro and Menu Editor**). Instead, these modifications can be done by manually editing the `macros.xml` file containing the macro definitions (located in the `Data` directory).

If some macros are deleted or modified by mistake, the default macro settings can be restored by pressing the **Restore Macro Settings** link (located in the lower left corner of the dialog). Alternatively, the `macros.xml` file can be replaced by a backed-up version containing the default settings (`macros.backup`). Both files are located in the `Data` directory.

The actual commands that can be used to define the macros are described in Appendix IX: Console and Macro Commands.

3.8 Clipping Planes

Clipping Planes allows you to change the clipping planes of the visualization window, i.e. how close and how far away objects are drawn. This can for example be useful if you want to visualize the interior of a protein or a ligand deeply buried inside a macromolecule.

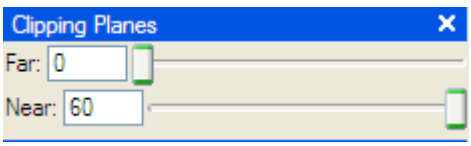


Figure 23: Clipping Planes dockable window.

Clipping Planes can be enabled by choosing **Window** | **Clipping Plane...** from the menu bar. Clipping Planes are enabled when the **Clipping Plane** window is shown and disabled when it is closed. Adjust the near and far slider until the desired region is shown.

3.9 Hiding Distant Residues

The **Hide Residues** dialog (see Figure 24) allows you to hide residues outside of a user-defined sphere: this can for example be used to show only the relevant residues near the binding site of the protein.

It is possible to set the center to of the sphere to the following objects if they are part of the workspace: The center of the protein(s), the center of the **Active Ligand**, the center of the **Reference Ligand**, the center of any cavity in the project, the center of the currently defined search space, or the center of a selection of atoms (if any). The residues are dynamically disabled or enabled when the **Sphere radius** slider is moved.

The lower pane of the **Hide Residues** dialog allows you to restrict the types of residues shown by toggling the appropriate button. If a given residue type is not contained in the sphere defined in the panel above, the button corresponding to the type will be grayed and can not be toggled.

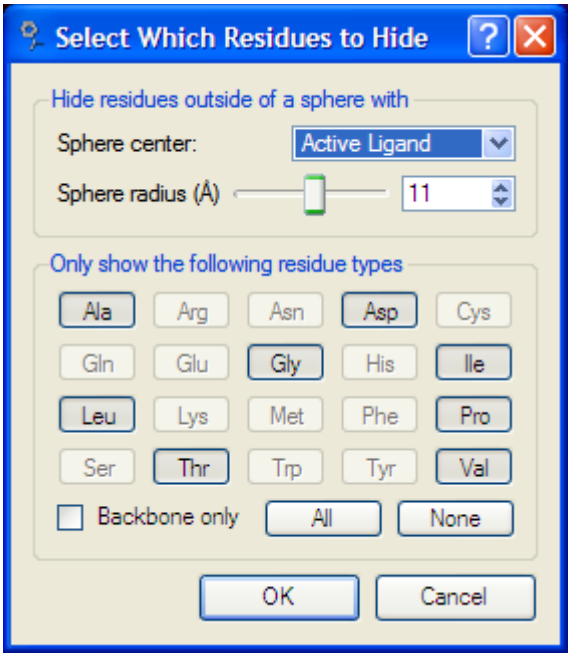


Figure 24: Hide residues dialog.

The **Backbone only** check box can be used to toggle whether side-chains are visible or not.

The **Hide Residues** dialog can be invoked by selecting **Window** | **Hide Residues Dialog**' or by pressing the **Hide Residues** button in the **MVD Toolbar**. In order to show all protein residues again, select the **Hide Residues** button on the **MVD Toolbar** (or choose **Window** | **Hide Residues Dialog** again).

3.10 Workspace Finder

The **Workspace Finder** located in the **MVD Toolbar** (see Figure 25) allows you to quickly search for molecule names and residue/atom IDs in the workspace. When a name or ID number (or part of it) is typed in the search box, the **Workspace Finder** will present a list of matches (a maximum of 30 matches is returned). It is also possible to search in atom coordinates by prepending the search with a '!' (e.g. searching for '!1.23' will return atoms where one of the coordinates starts with 1.23).

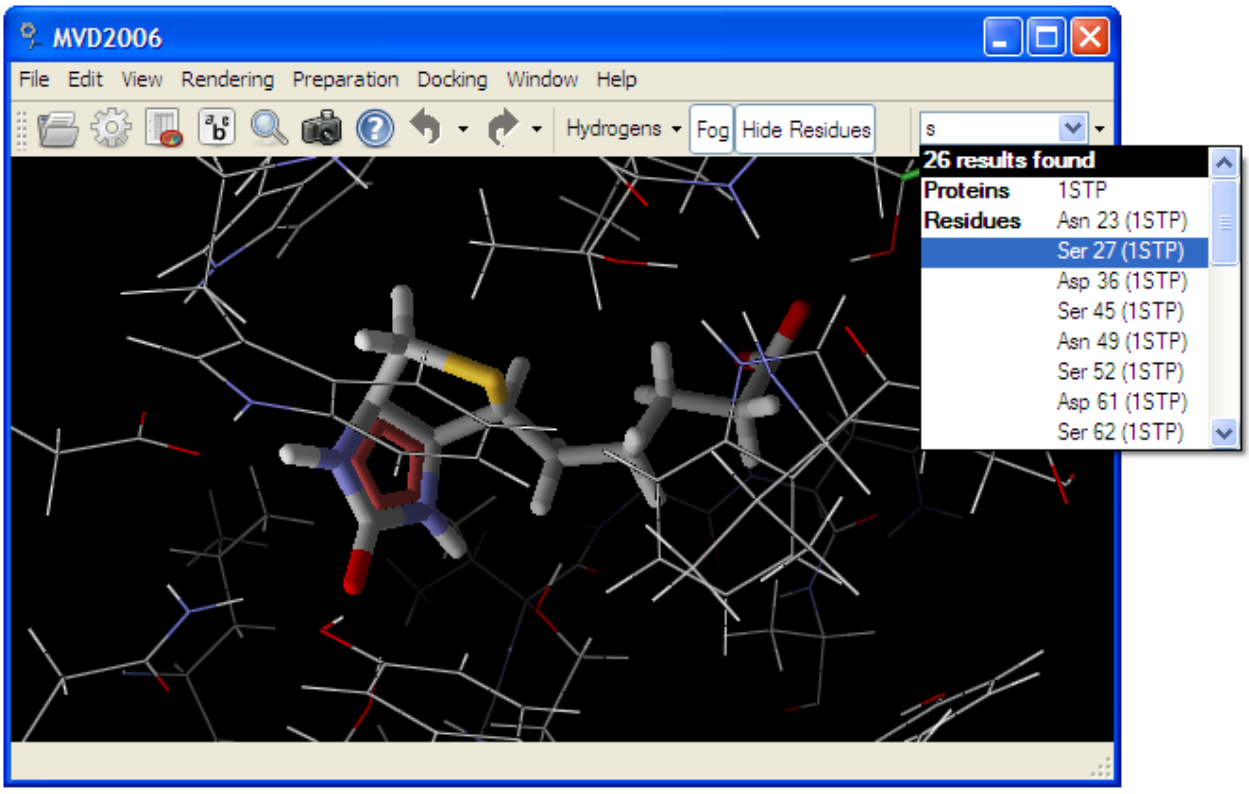


Figure 25: Workspace Finder dialog.

By default, the **Fit to screen** option is enabled so that items (molecules, residues, or atoms) are fitted to the **Visualization Window** while browsing the list of results found. The **Fit to screen** option can be disabled in the options panel invoked by pressing the small button on the right hand side of the **Workspace Finder** search box.

The **Workspace Finder** is invoked by typing characters in the search box (text field) located in the far right side of the **MVD Toolbar**. A result is selected by pressing the **Return** key. Pressing the **Escape** (Esc) key or mouse-clicking outside the **Workspace Finder** window will cancel the current search query.

3.11 Sequence Viewer

The **Sequence Viewer** dialog (see Figure 26) allows you to inspect protein residues in an easy manner. Using the context menu on the **Sequence Viewer** window it is possible to select residue atoms in the **Visualization Window**, hide non-selected residues, change between one and three-letter residue names, and toggle details about secondary structure.

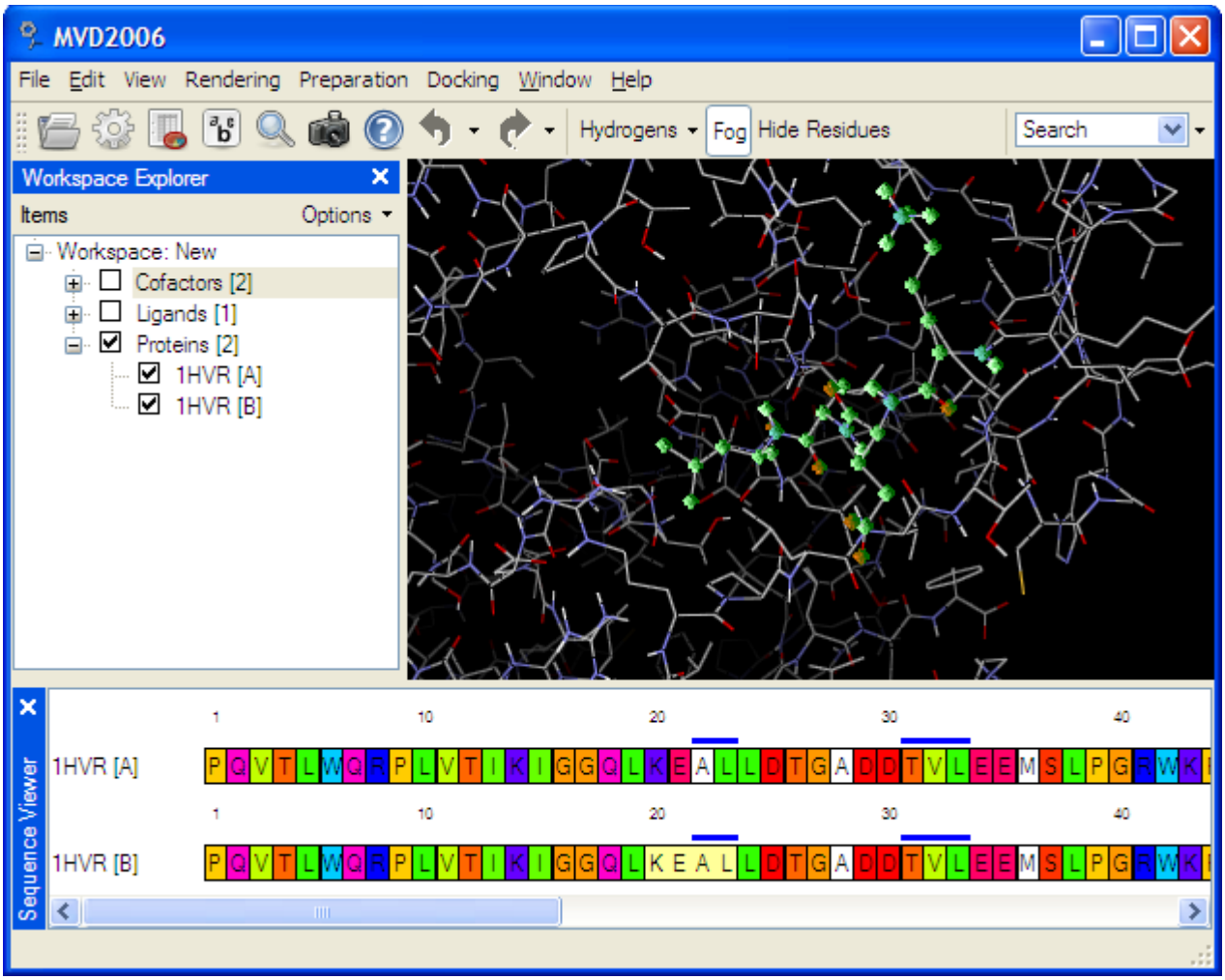


Figure 26: Sequence viewer with selection of four residues highlighted in the Visualization Window.

The **Sequence Viewer** dialog can be invoked by selecting **Window** | **Sequence Viewer**' or using the **Ctrl-Shift-S** keyboard shortcut.

3.12 Workspace Properties

Workspaces can contain user-specified notes. Further, the title of the workspace can be changed using the **Workspace Properties** dialog. The **Workspace Properties** dialog can be found in the **Edit Properties...** context menu on the **Workspace** item in the **Workspace Explorer** or via **Edit** | **Workspace Properties...** and **Window** | **Workspace Properties** (see Figure 27).

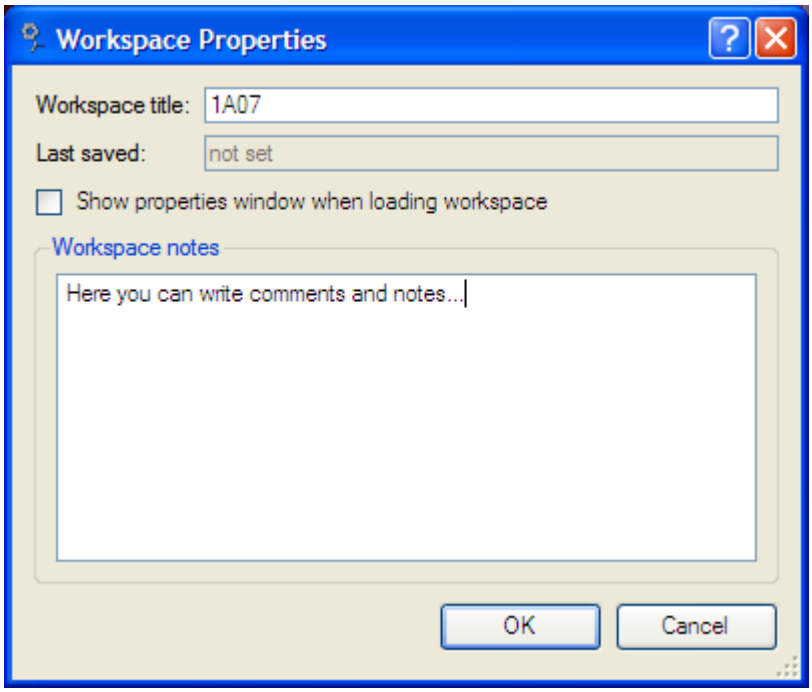


Figure 27: Workspace Properties dialog.

3.13 Measurements and Annotations

Distances and angles can be measured directly in the 3D world (see Figure 28).

If two atoms are selected, the distance between them will be shown in the **Properties Window**.

If three connected atoms are selected, the angle that they span will be shown in the **Properties Window**.

If no atoms are selected, and a bond is highlighted, the field **Torsion Angles** in the **Properties Window** will show the torsion angle(s), defined through this bond.

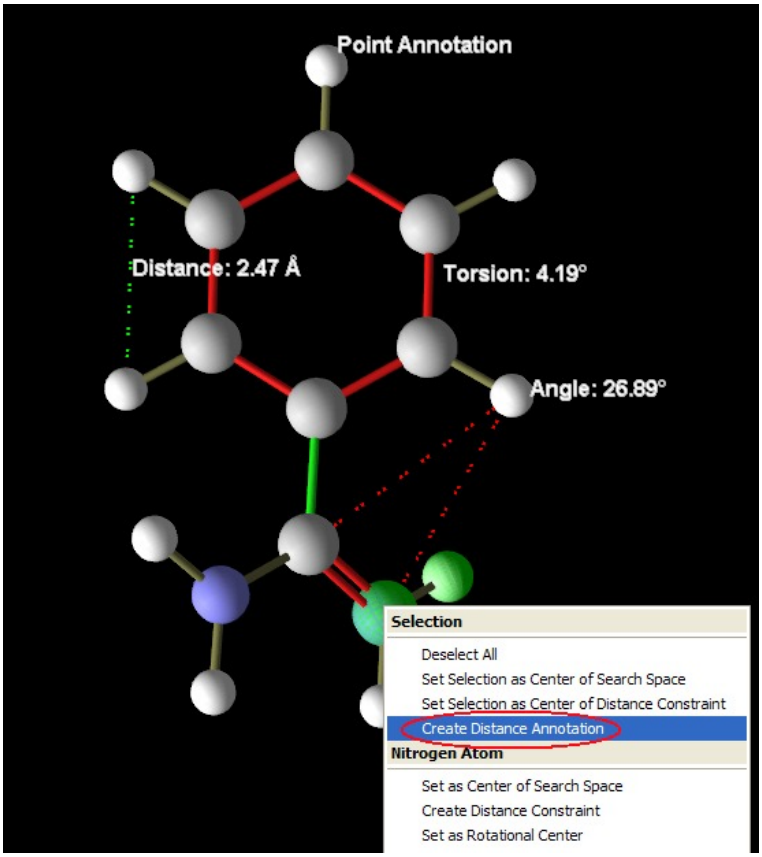


Figure 28: Annotations and measurements.

Measurements can also be made permanent as annotations. There are different kinds of annotations. To create annotations, select 1-4 atoms and use the context menu (right-click mouse button) and choose **Create ... Annotation**. The text can be edited before the annotation label is created. Annotations are added to the **Workspace Explorer** category: **Annotations**

3.14 Creating Labels

To create labels use the **Create Label** dialog, which can be invoked via **Create Labels...** in the **Workspace Explorer** context menu (on molecular categories: **Proteins**, **Ligands**, and **Poses**) or via the **Window** | **Labels** menus.

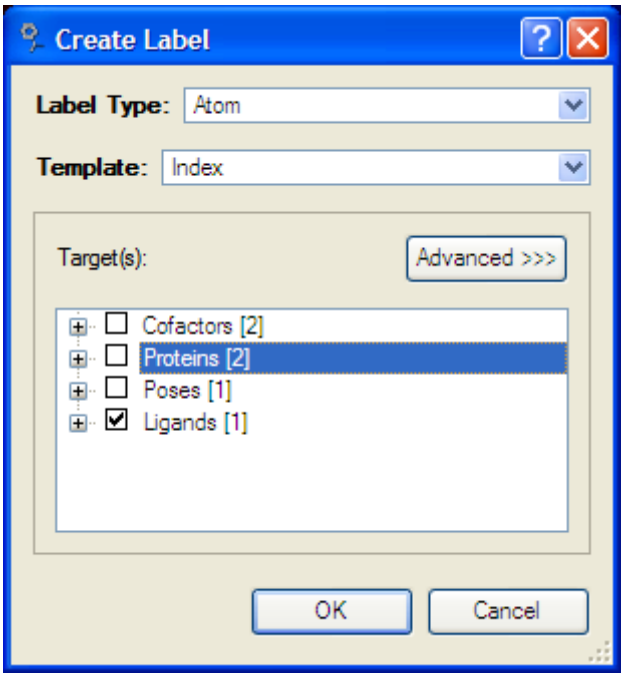


Figure 29: Creating a new label.

The **Create Label** dialog makes it possible to label different *object levels*: atoms, bonds, molecules, residues or torsion trees. The labels can be chosen from a list of standard templates or constructed from a list of available variables (using the **Advanced** tab).

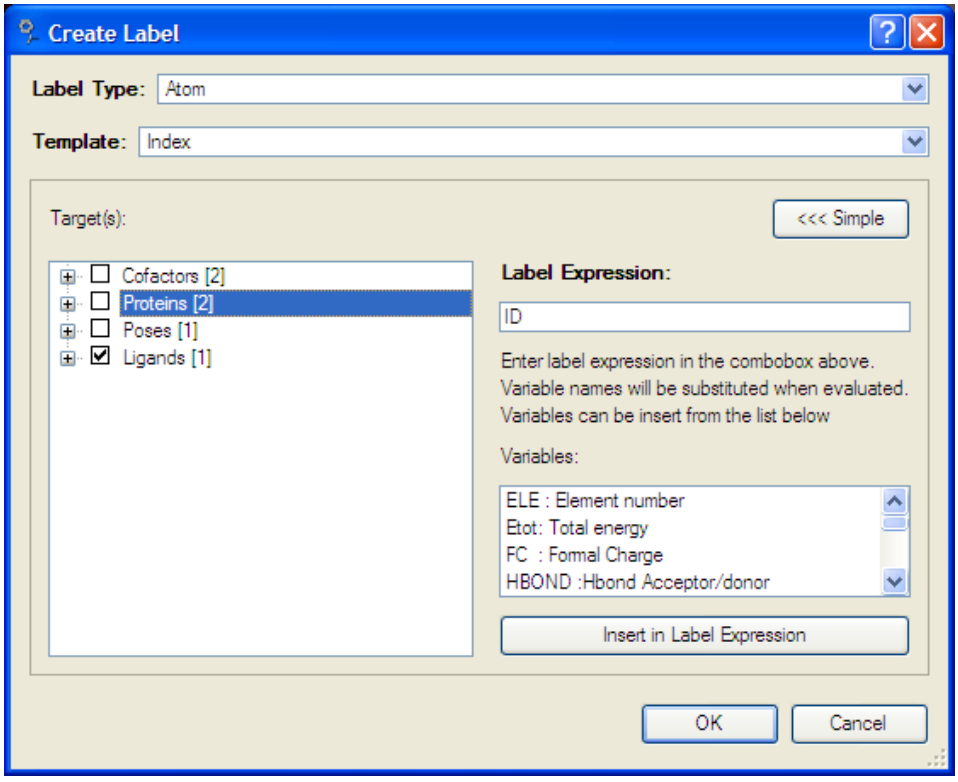


Figure 30: Advanced label expression dialog.

Labels will occur in the **Labels** category in the **Workspace Explorer** and can be removed or hidden using the context menu or by pressing the labels tool bar button.

3.15 Creating Molecular Surfaces

Surfaces can be created for all molecular objects, and subsequently customized. Support for **Solvent** or **Surface Accessible Surfaces**, coloring by **Hydrophobicity**, **Electrostatic Potential**, or **Solid Color**. Surfaces can be drawn transparently, as dots, lines, or solid polygons.

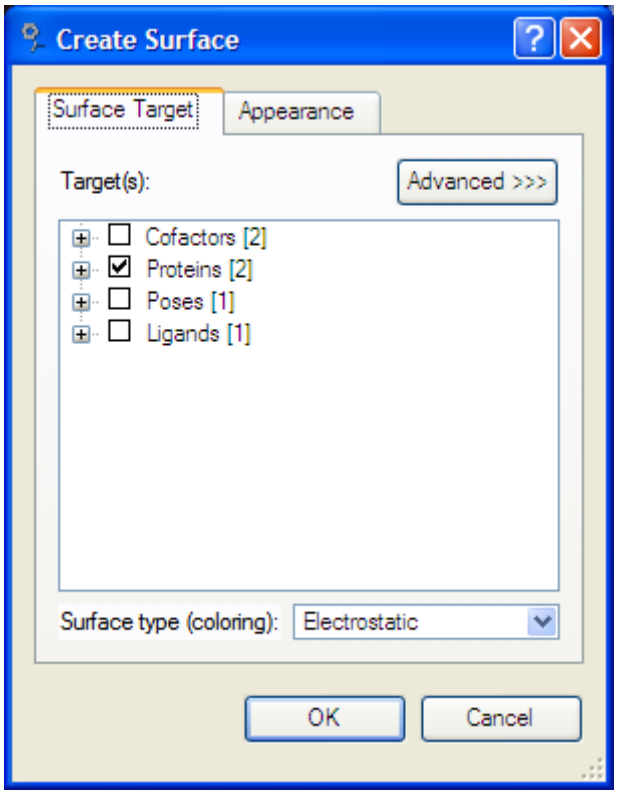


Figure 31: Creating a new surface.

Surfaces can be created via **Create Surface...** from the context menu in the **Workspace Explorer** or via **Window** | **Surfaces**.

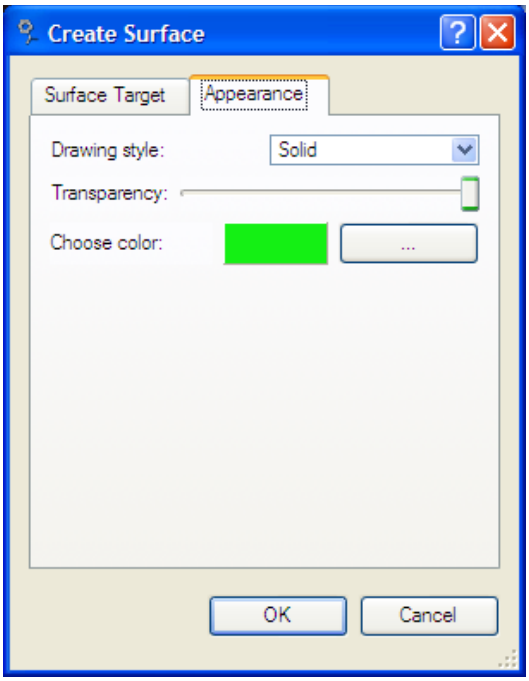


Figure 32: Changing surface appearance.

3.16 Creating Protein Backbones

The backbone of the protein can be visualized by using the **Create Backbone** dialog. The dialog can be invoked by using the context menu on the **Proteins** category (or a single Protein item) in the **Workspace Explorer**.

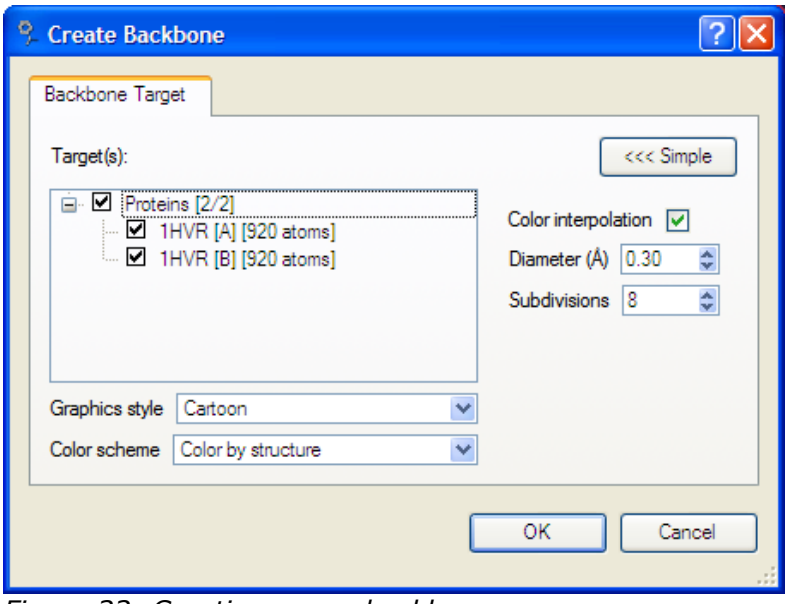


Figure 33: Creating a new backbone.

The **Create Backbone** dialog allows you to select which proteins (or protein chains) the backbone should be created for.

Two main graphics styles can be used. The **Cartoon** style visualizes the secondary structure of the protein(s) using arrows to represent beta sheets and helical lines for alpha helices (see Figure 34).

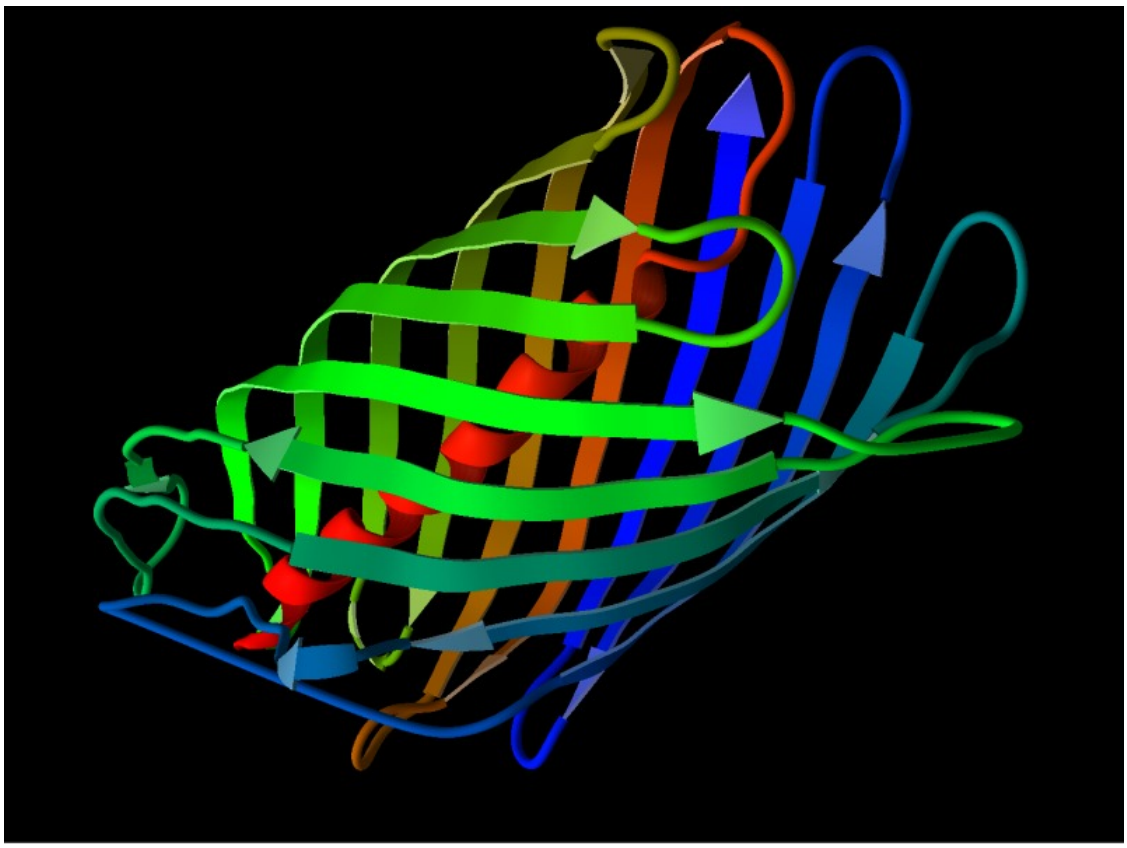


Figure 34: Cartoon graphics style.

If the **Tube** graphics style is used, the backbone is visualized as a spline (a piecewise parametric polynomial curve) interpolating the positions of the alpha carbons in the backbone (see Figure 35).

Figure 35: An example of a protein backbone using the Tube graphics style.

It is also possible to set the color scheme for the backbone. **Color by structure** colors the backbone based on the secondary structure information (alpha helices are colored yellow, beta sheets are colored blue, and coil is colored gray). **Color by residue position** colors the backbone based on the residues order of occurrence creating a rainbow color effect. **Color by chain** colors each individual protein chain in a different color. **Color by atom** colors the backbone by using the currently shown color of the protein backbone atoms (the color used is taken from the C-alpha atom).

On the advanced panel, the **Color interpolation** check box allows you to determine whether the backbone color should be interpolated between the atoms it passes through or should be held constant between atoms. **Diameter (Å)** sets the width of the backbone in angstrom, **Subdivision** sets the resolution of the backbone (the number of subdivisions between each residue in the protein).

Backbones appear in the **Backbones** category in the **Workspace Explorer** and can be removed via the context menu or hidden using the check box.

3.17 Making Screenshots

Screenshots can be made by choosing **Window** | **Capture Screen**.

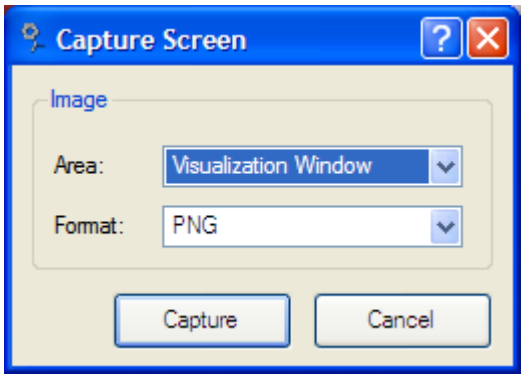


Figure 36: Screen Capture dialog.

It is possible to specify whether to capture the **Visualization Window** only (the 3D view) or the entire **Desktop** (see Figure 36). The captured region can be saved in JPG, BMP, or PNG file format.

4 Preparation

4.1 Import of Molecules

Molecules can be imported into MVD using the **Import Molecule...** menu option located in the **File** menu. A shortcut is provided from the tool bar by clicking on the **File** folder icon or using the **Ctrl-O** keyboard shortcut. Molecules can also be imported by dragging-and-dropping the molecular file into the main application window.

Currently, MVD supports the following file formats:

- Protein Data Bank (pdb/ent)
- Sybyl Mol2 (mol2)
- MDL (sdf/sd/mol/mdl)

Notice that only PDB and Mol2 files can contain proteins, cofactors, and structural waters. In general, it is recommended to use Mol2 or SDF files for ligands since they (usually) contain bonding information.

From the **Import Molecules** dialog shown in Figure 37, it is possible to select which molecules to import, prepare molecules, and inspect warnings found during parsing of the imported file.

Notice: If more than 10 ligands are present in the file (typically SDF or Mol2 files), a subset of the ligands can be selected for import using the **Specify ligand range** option (See Figure 37). Since it is computationally slow to display a large number of molecules (e.g. thousands of compounds), ligands and poses are not automatically shown in the **Visualization Window** if the number of molecules imported exceeds 50 (for each category).

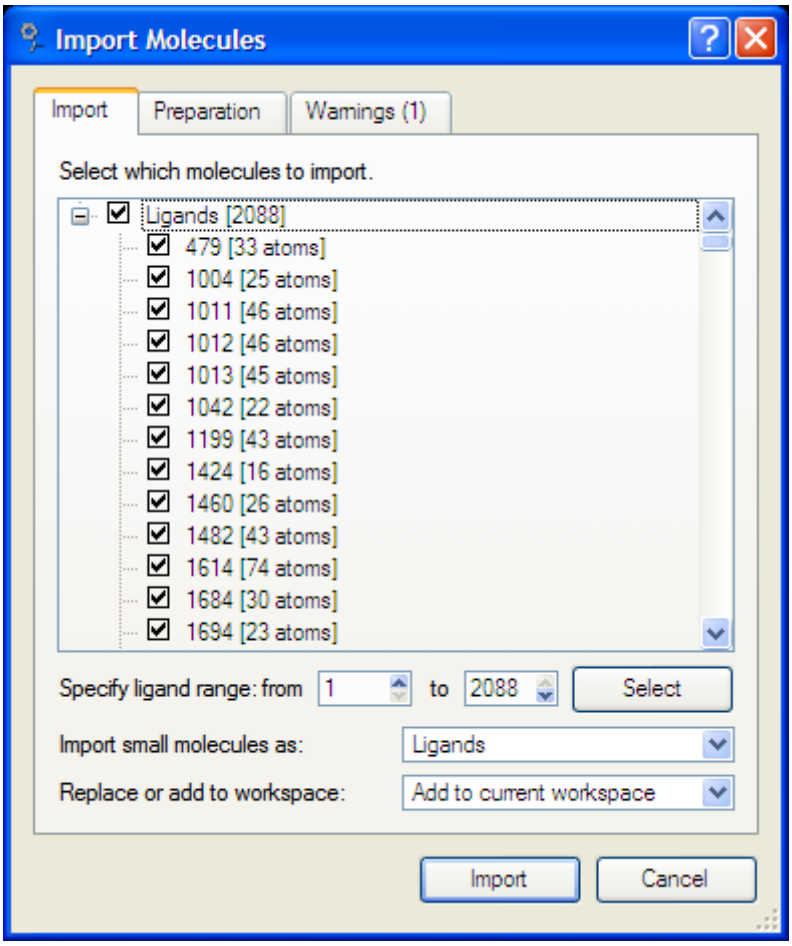


Figure 37: Import Molecules dialog.

When all relevant molecules have been imported (a minimum of one protein and one ligand is required), the molecules can be automatically prepared (see next section).

4.2 Automatic Preparation

Some molecular file formats support information about bond type and charge (e.g. Mol2) while others do not (e.g. PDB). It is important for MVD to dock correctly, that the structures have been properly prepared. That is, that the atom connectivity is known and that the correct bond order and charges have been assigned.

The **Prepare Molecules** dialog allows the user to perform the necessary preparation. It is invoked automatically when importing Mol2, SDF, or PDB files, and can be invoked manually by selecting **Preparation** | **Prepare Molecules** or by using the context menu (e.g. **Prepare Ligand...**) on molecules in the **Workspace Explorer**.

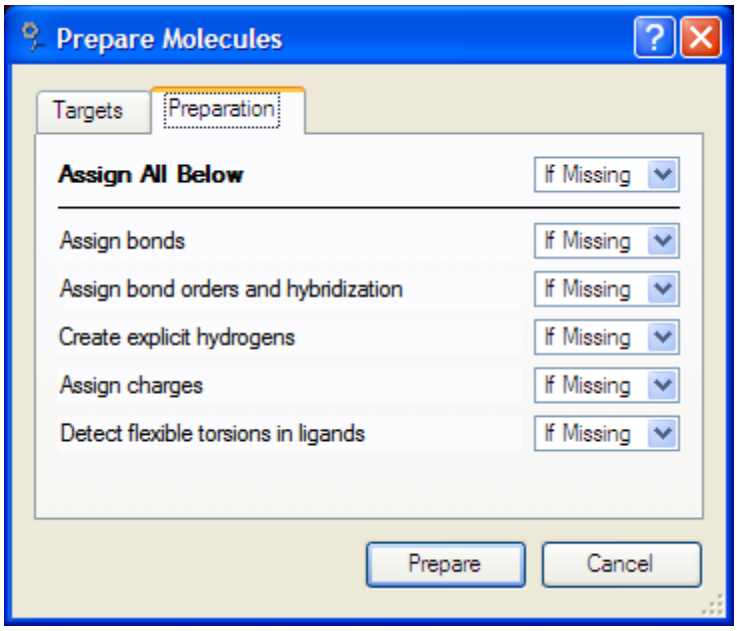


Figure 38: Preparing molecules.

Within all preparation types the following four different possibilities are available (see Figure 38):

- **Always**. Unconditionally performs the preparation by MVD.
- **Never**. Skips the preparation.
- **If Missing**. The preparation will only be performed if no knowledge is already present (e.g. if bond orders exist in the Mol2 file, bond orders are not assigned by MVD. However, if bond order information is not included – MVD will assign it).
- **Remove**. Tries to remove preparation (e.g. for bond orders - all bonds are set to be single bonds, for explicit hydrogens - all hydrogens are removed).

Notice: The preparation options (Always, Never, If Missing, Remove) applies to each individual molecule (not each individual bond or atom). For instance, setting 'Assign bonds' to 'If Missing' results in covalent bonds being created for molecules not containing any bonds at all while molecules with bond information will preserve their bond assignments. Likewise, setting 'Create explicit hydrogens' to 'If Missing' will not add additional hydrogens to molecules containing e.g. polar hydrogens only. In this case, 'Always' should be used if all hydrogens should be created.

Assign Bonds

This option allows to determine which atoms are connected (covalently bound). Two atoms are connected if their distance is more than 0.4Å and less than the

sum of their covalent radii plus a threshold of 0.45Å (the threshold is set to 0.485Å if one of the atoms is Phosphorus).

Assign Bond Order and Hybridization

This options allows recognition of bond orders (whether bonds are single, double or triple, ...), the number of hydrogens attached to the atoms, and their hybridization (SP, SP2, SP3). Also aromatic rings will be detected. It should be noted that this assignment is not always perfect - different protonation states can be difficult to assign properly. A detailed description can be found in Appendix VII: Automatic Preparation.

Notice: The algorithm only assigns the number of implicit hydrogens to each atom. No actual atoms will be added. The next option **Create explicit hydrogens** allows you to add explicit hydrogens based on the implicit ones.

Create Explicit Hydrogens

Creates hydrogens matching the predicted number of hydrogens in the step above. The hydrogens are placed according to geometric criteria (i.e. SP3 hybridized atoms are kept at a 109 degrees geometry). The hydrogens are placed at standard distances according to the atom they are connected to. No energy minimization is performed.

Assign Charges

This option allows to assign partial charges to each atom based on the scheme described in Appendix I: Docking Scoring Function.

Detect Flexible Torsions In Ligands

This option determines which bonds that should be considered flexible during docking. It is advisable always to set this option to either **If Missing** or **Always**. If this option is set to **Remove**, the ligand will be considered rigid during docking.

Hydrogen Bonding Type

Atom hydrogen bonding types (acceptor, donor, both or non-polar) are always set during preparation.

4.3 Manual Preparation

Molecules can be manually prepared using the context menus of highlighted atoms or bonds (see below).

Set Hybridization

Hybridization (SP, SP2, SP3) can be manually assigned to atoms by right-clicking on the atom in question and selecting the **Set Hybridization** menu option.

Set Hydrogen Bond Type

Hydrogen bond type (donor, acceptor, both, non-polar) can be manually assigned to atoms by right-clicking on the atom in question and selecting the **Set Hydrogen Bond Type** menu option.

Set Hydrogen Count

The **Set Hydrogen Count** menu option can be used to set the number of explicit hydrogens attached to the highlighted atom.

Assign Charges

Currently, the MVD scoring function (MolDock Score, see Appendix I: Docking Scoring Function for more details) uses partial charges assigned when running the **Preparation** dialog. However, the assignment of charges is based on standard templates and charge assignments can be missing in some cases. It is possible to manually assign partial charges to atoms by right-clicking on the atom in question and selecting the **Set Partial Charge** menu option.

Set Bond Order

Bond orders can be manually assigned by right-clicking on the bond in question and selecting the **Set Bond Order** menu option.

Notice that bonds are not visible in some visualization styles. The most suitable view is the ball-and-stick style, which can be set from the **Rendering** menu in the menu bar.

Set Ligand Flexibility

Flexible torsions in the ligand can manually be set rigid or flexible by right-clicking on a bond and selecting the **Set Flexibility** menu option.

Set Root Atom

When automatically detecting and assigning flexible torsion angles (using the automatic preparation procedure), a root atom is chosen. The root atom is used as root in the torsion tree, which is used to construct the ligand conformation during the docking process. Sometimes, the docking performance

can be improved by choosing another atom to be the root atom. To manually set the root atom, right-click on an atom and select the **Set as Root Atom** menu option.

Notice that bonds are not visible in some visualization styles. The most suitable view is the ball-and-stick style, which can be set from the **Rendering** menu in the menu bar.

4.4 Protonation Wizard

This wizard highlights residues likely to have a non-standard protonation state (such as histidine) and makes it easy to change the protonation state via the context menu. To invoke the wizard choose **Preparation** | **Protonation Wizard**.

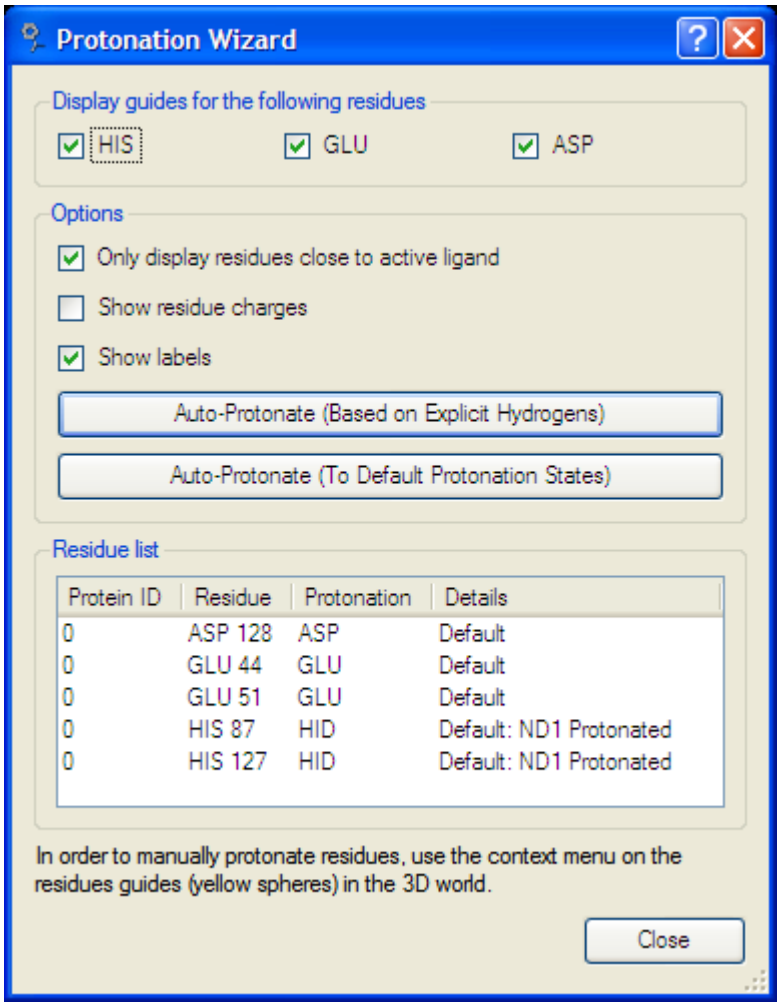

Protonation Wizard
Display guides for the following residues
HIS
GLU
ASP
Options
Only display residues close to active ligand
Show residue charges
Show labels
Auto-Protonate (Based on Explicit Hydrogens)
Auto-Protonate (To Default Protonation States)
Residue list
Protein ID
Residue
Protonation
Details
0 ASP 128 ASP Default
0 GLU 44 GLU Default
0 GLU 51 GLU Default
0 HIS 87 HID Default: ND1 Protonated
0 HIS 127 HID Default: ND1 Protonated
In order to manually protonate residues, use the context menu on the residues guides (yellow spheres) in the 3D world.
Close

Figure 39: Protonation Wizard.

The top panel (**Display guides for the following residues**) chooses which residues to display in the 3D view.

The **Options** panel can be used to toggle whether only residues close to the active ligand should be displayed, and whether charges and labels should be displayed or not.

Two buttons make it possible to perform automatic protonation. The first **Auto-Protonate (Based on Explicit Hydrogens)** can be used if the protein has been prepared in another program, and the hydrogens have been explicitly set. This method will assign the protonation states only taking into account the positions of explicit hydrogens in the protein.

The other button **Auto-Protonate (To Default Protonation State)** assigns a standard protonation (corresponding to neutral pH), i.e. histidine is protonated at the delta-nitrogen.

The last panel (**Residue list**) shows a list of residues, and their current protonation state. If a residue is selected on the list, the 3D view will focus on it.

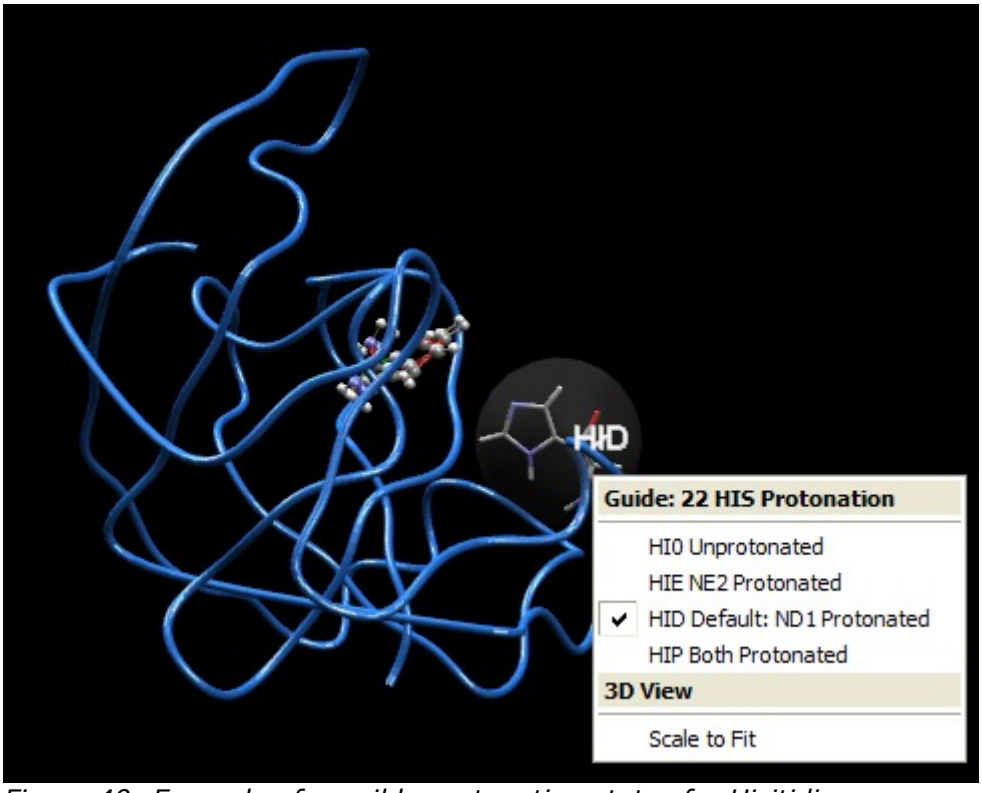


Figure 40: Example of possible protonation states for Hisitidine.

In the 3D world, all relevant residues are displayed with a green sphere around them (if the protonation state is erroneous or not supported, the spheres will be red). By using the context menu on these *Protonation Guides*, an alternative protonation can be set.

The protonation states are important as they determine the hydrogen bonding capabilities of the protein, and hence will affect the poses found, and the docking energies.

5 Docking Functionality

5.1 Cavity Prediction

Potential binding sites (also referred to as cavities or active sites) can be identified using the built-in cavity detection algorithm (see Appendix III: Cavity Prediction for details).

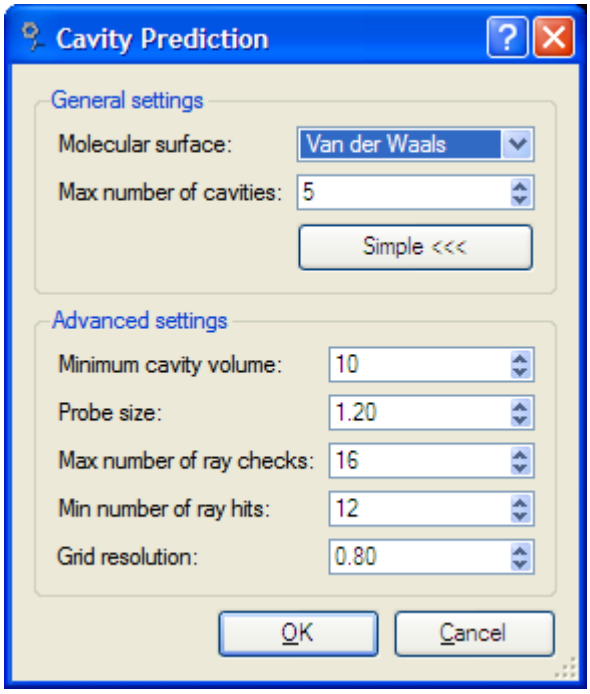


Figure 41: Cavity Prediction dialog.

A dialog is available for detecting cavities (see Figure 41), allowing to customize the sensibility (and type) of cavity search. The **Cavity Prediction**

dialog can be invoked via the context menu in the **Workspace Explorer** (in the **Proteins** category) or by selecting the **Detect Cavities** menu option from the **Preparation** menu.

Cavities found are listed in the **Workspace Explorer** in the **Cavities** category. Visualization of the cavities can be toggled on and off. Moreover, volume and area are listed for each cavity.

Notice: If no cavities are identified, ligands can only bind to the surface of the protein (or the cavity is too small to be detected). This situation makes it more difficult for the docking engine to identify the correct binding modes.

5.2 Constraints

Constraints are limitations imposed on the molecular system based on chemical insight or knowledge. Constraints can dramatically increase docking accuracy and speed, as they often limit the search space considerably.

There are two fundamental kinds of constraints:

- Hard Constraints: The docking engine tries to fully satisfy these constraints, i.e. a hard constraint could be used to force a specific ligand atom to be in a given region. The docking engine will enforce these constraints by moving or modifying the poses during docking. If several hard constraints exist, and none of them are satisfied, the system will choose to satisfy an arbitrary one. Notice that this means that not necessarily all hard constraints are satisfied. *If a hard constraint is not satisfied, it will add 100 units to the soft constraint energy penalty*.
- Soft Constraints: These act as extra energy terms, and contribute to the overall energy of the system. As such, they can be more or less satisfied. They can for example be used to reward ligands in certain regions. If several (enabled) soft constraints exist in the workspace, they are ALL taken into account (i.e. several extra terms are added to the overall docking energy function while docking).

Creating Constraints.

Distance constraints constrain ligand atoms to a given position in 3D-space (see Figure 42). They are used to constrain some or all atoms of a ligand to the vicinity of this position. Distance constraints are visualized as an inner and outer sphere where some ligand atoms must be present between the spheres.

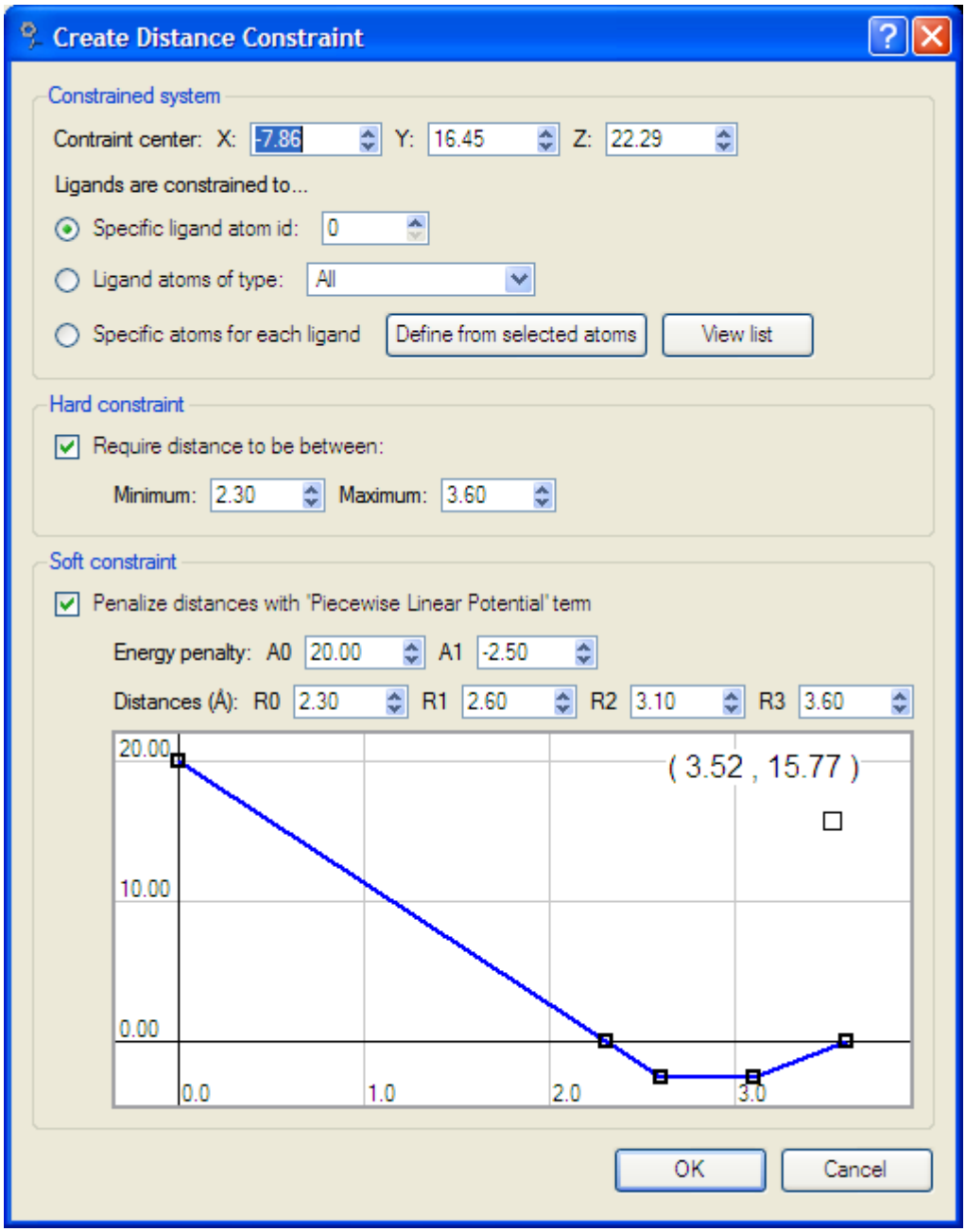


Figure 42: The Distance Constraint dialog.

The **Distance Constraint** dialog can be invoked either via the context menu on an atom (**Create Distance Constraint**), or by selecting one or more atoms, and using the context menu option (**Set Selection as Center of Distance Constraint**). If several atoms are chosen, their mean position will be set as center for the constraint.

The top panel allows the user to modify the location of the constraint (**Constraint Center**). It also controls which parts of the ligand should be constrained. Either a single atom (the **Specific Ligand Atom (ID)** option) or multiple atoms (the **Ligand Atoms of Type** drop down menu). The different choices for multiple atoms are: **All** (meaning all atoms), **None** (which causes

the constraints to try to remove atoms within the constraint range), **Hydrogen Donor**, **Hydrogen Acceptor**, **Hydrogen Donor or Acceptor (Both)**, **Non-polar**, and **Ring Atoms** (atoms in aromatic or aliphatic rings). Additionally, it is also possible to specify ligand atoms from a current selection of atoms using the **Specify atoms for each ligand** option and pressing the **Define from selected atoms** button. This also applies to more than one ligand, which makes it easier to constrain specific atoms for a set of ligands present in the workspace. The **View list** button can be used to inspect the current set of selected ligand atoms.

A **Hard constraint** (see above) range can be specified. If this is enabled the docking engine will try to force the selected ligand atoms to be within this range.

The bottom panel (**Soft constraint**) allows the user to specify a specific potential applied to the selected ligand atoms. The potential is a piece-wise linear potential, which is the same type as used in the docking score function (see Appendix I: Docking Scoring Function). It is shown graphically in the graph at the bottom.

When applying soft constraints the following procedure is used: for all chosen ligand atoms (as defined by the **Ligands atoms of Type** or **Specific ligand atom** input fields) the distance between the center of the constraint and the atom is calculated. The potential is then evaluated for all these distances, *but only the lowest energy is returned as the soft constraint energy.* That is, only the atom with the lowest energy relative to the constraint potential is taken into account. The reason for this is, that if you for example want to reward ligands with a hydrogen acceptor close to hydrogen donor in the protein, it does not make sense to punish other atoms in the vicinity of the constraint if one hydrogen acceptor is already at its optimal distance from the donor.

Ligand Atom Constraints

Another type of constraint is the **Ligand Atom Constraint**. It is used to constrain a specific ligand. Since **Ligand Atom Constraints** are defined using a list of atom IDs, they are specific to ligands and are only applied to the ligand on which they are defined.

To create a **Ligand Atom Constraint**, select a number of atoms (in the same ligand) in the **Visualization Window**. Ensure that no other objects are selected, and choose **Constraint Selected Ligand Atoms** from the context menu (right-click mouse button). It is also possible to use the context menu on a single ligand atom (**Create Ligand Atom Constraint**) without performing a selection.

The **Ligand Atom Constraint** dialog will appear (see Figure 43). It is possible to modify the list of atoms in the ligand by entering a comma-separated list of IDs.

Notice: **Ligand Atom Constraints** are always *soft constraints*. It is possible to choose whether the chosen atoms in the ligand should be *rewarded* or *penalized* for contacts with the target molecules (proteins, cofactors and water).

The criteria for contact used here is purely based on the distance between the chosen ligand atoms and the closest atom in any target molecule. The distance threshold for defining contacts can also be customized (using the **Define atom contact threshold (Å)** input field).

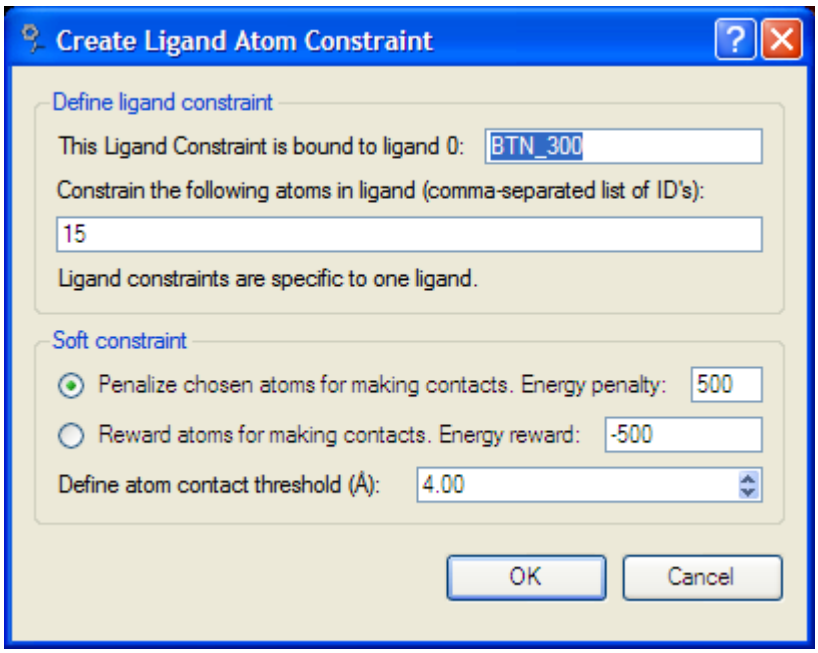


Figure 43: Ligand Atom Constraint dialog.

Typical Uses

Constraints are useful if something about the system is known in advance. If perhaps a hydrogen bond from a hydrogen donor was known to be present – a distance constraint could be set up at the position of the protein hydrogen donor, and a hard constraint could force hydrogen acceptors in the ligand to satisfy the hydrogen bond.

5.3 Docking Wizard

When all the molecules have been prepared, the docking can commence. To start the **Docking Wizard**, select **Docking** | **Docking Wizard**. A shortcut is provided by clicking on the docking icon (gear wheel) on the tool bar. Additionally, the keyboard shortcut **F1** is available.

Notice: In order to initiate the docking, at least one protein and one ligand molecule have to be present in the workspace.

Choose Which Ligands to Dock

The first action is to choose which ligands to dock (see Figure 44).

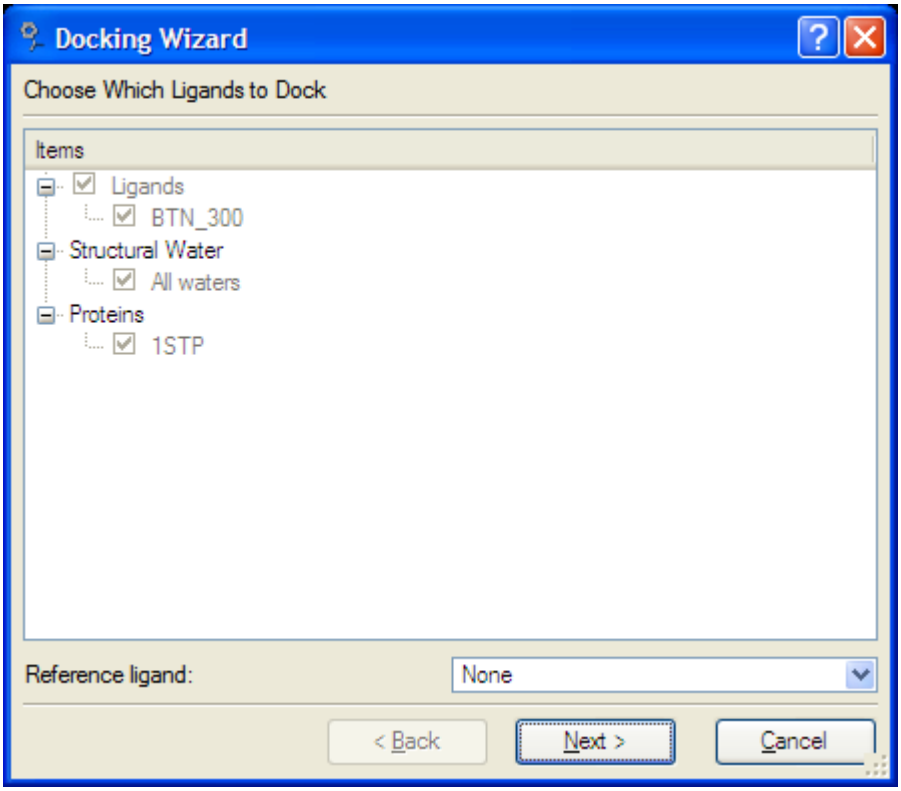


Figure 44: Select which ligands to dock.

If more than one ligand is available the user can select which ones to use by clicking on the corresponding molecules in the window. If more than one ligand is selected, all selected ligands will be docked one at a time. Structural waters and cofactors (if any) are always included in the docking simulation (remember to remove them from the workspace if they should not be included). Moreover, a reference ligand can be specified at the bottom. The reference ligand is used to calculate the root-mean-squared deviation (RMSD) between the reference ligand and the docked pose. The reference ligand - or ligands - are only available if they are compatible (w.r.t. symmetry, identical number of heavy atoms, etc.) with the ligands selected for docking.

Notice: If more than 10 ligands are present in the workspace, a subset of the ligands can be selected for docking using the **Specify ligand range** option (not shown on Figure 44).

Choose Scoring Function and Define Binding Site

MVD includes *MolDock Score* [THOMSEN 2006] and a grid-based version called *MolDock Score [Grid]* for evaluating docking solutions. The *MolDock Score* is further described in Appendix I: Docking Scoring Function. The *MolDock Score*

[Grid] is identical to the *MolDock Score* except that hydrogen bond directionality is not taken into account. The grid-based scoring function provides a 4-5 times speed up by precalculating potential-energy values on an evenly spaced cubic grid (see Appendix:Appendix XII: MolDock Score [Grid] for more details).

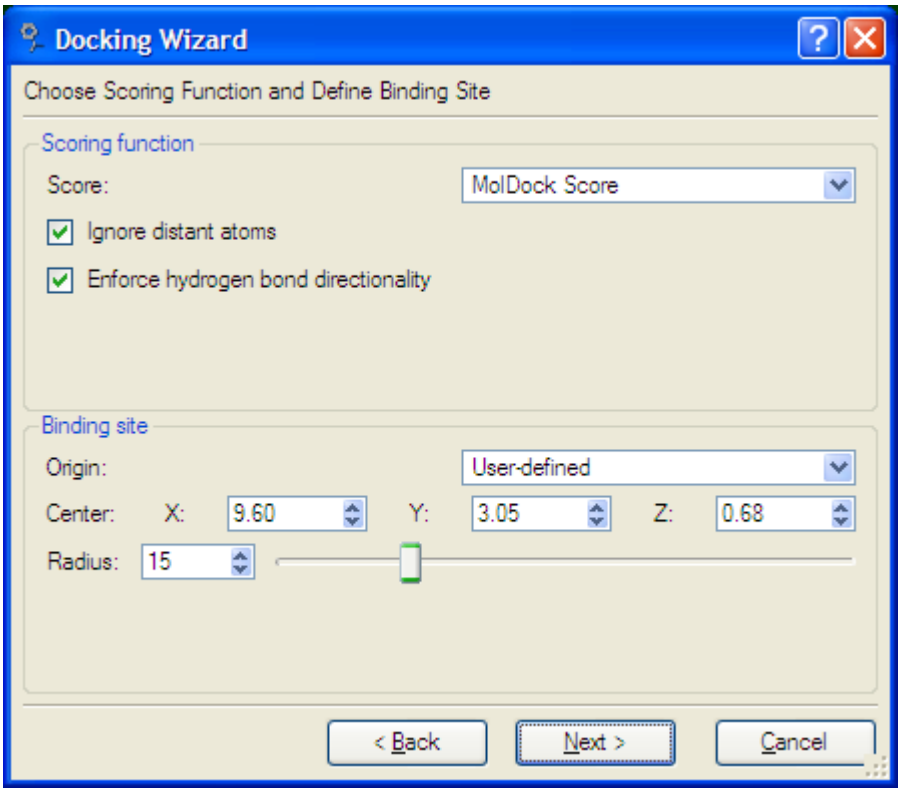


Figure 45: Choosing scoring function.

The **Ignore distant atoms** option is used to ignore atoms far away from the binding site. Thus, atoms more than **Radius** angstroms away from the center of the binding site are ignored in the scoring function. This reduces the overall computing time significantly when working on large molecules. Notice that charged atoms (capable of long-range interactions) are always taken into account in the scoring function.

The **Enforce hydrogen bond directionality** option is used to check if bonding between potential hydrogen bond donors and acceptors can occur. If hydrogen bonding is possible, the hydrogen bond energy contribution to the docking score is assigned a penalty based on the deviations from the ideal bonding angle. Using this option can significantly reduce the number of unlikely hydrogen bonds reported.

For the grid-based scoring function, the **Grid resolution** option (not shown in Figure 45) can be used to set the granularity of the generated energy grids.

The **Binding site** specifies the region of interest and thus where the docking

procedure will look for promising poses (ligand conformations). The **Origin** determines which area of the protein is expected to include the binding site.

If cavities have been identified the user can pick one of these as the preferred area of interest. Further, if a reference ligand is being used, the center of the reference ligand can be used. By default (if no cavities have been identified and no reference ligand is specified), the center of the bounding box spanning all protein(s) will be used. The actual center of the binding site used is listed in the **X**, **Y**, and **Z** boxes in the window. Besides the center of the binding site, a **Radius** can be specified (default is 15 angstrom). The **Search Space** region will be shown in the **Workspace Explorer** in the **Constraints** category.

Notice: A sphere in the **Visualizer Window** indicates the position and size of the current search space region (see Figure 46).

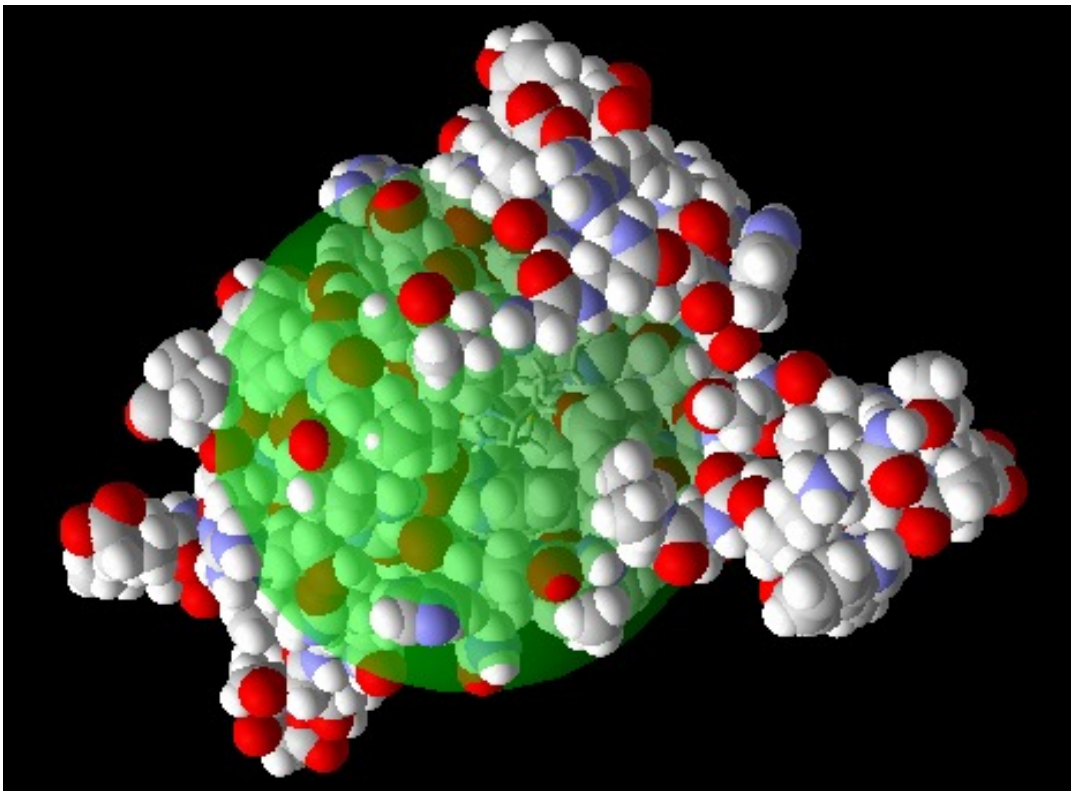


Figure 46: Example of search space region (green sphere).

Enable or Disable Additional Constraints

If constraints (besides the search space region) have been added to the workspace, they can be toggled on and off in the **Enable or Disable Additional Constraints** tab (see Figure 47). In order for a constraint to be meaningful it must be defined within the current search space region.

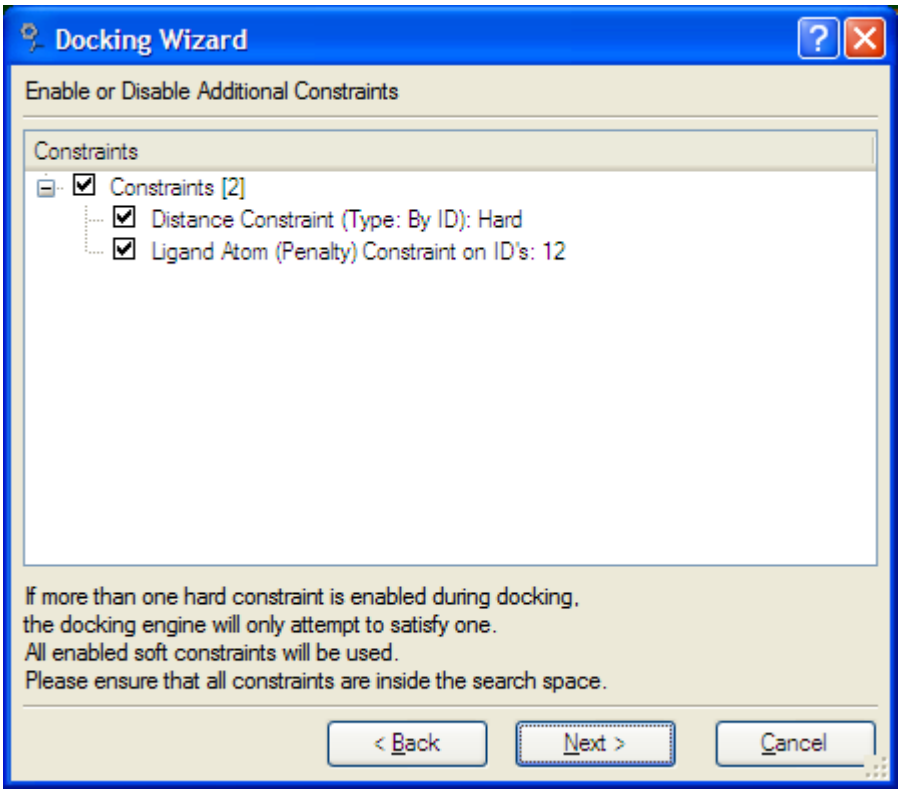

Docking Wizard
Enable or Disable Additional Constraints
Constraints
Constraints [2]
Distance Constraint (Type: By ID): Hard
Ligand Atom (Penalty) Constraint on ID's: 12
If more than one hard constraint is enabled during docking,
the docking engine will only attempt to satisfy one.
All enabled soft constraints will be used.
Please ensure that all constraints are inside the search space.
< Back
Next >
Cancel

Figure 47: Enabling or disabling user-defined constraints.

Choose Search Algorithm

The default search algorithm used in MVD is the *MolDock Optimizer* [THOMSEN 2006]. From MVD v1.5, an alternative search algorithm named *MolDock SE* (simplex evolution) is available. MolDock SE perform better on some complexes where the standard MolDock algorithm fails. This is usually the case when the ligand has lots of internal degrees of freedom (many torsion angles). The MolDock SE algorithm (and its parameters) is further described in Appendix XI: MolDock SE.

Notice: MolDock SE is still experimental, as all parameters has not been fine-tuned yet. Although it should be stable, it is still to be considered a technology preview version.

The **Number of runs** specifies the number of times that the docking simulation is repeated for each ligand chosen to be docked. Sometimes more than one run is needed to identify promising poses (in particular for ligands having more than 15 flexible torsions or if no promising cavities exist). If cavities have been identified (see Section 5.1) the poses found by the search algorithm can be constrained to the region spanned by the cavity (by using the **Constrain poses to cavity** option). This option greatly reduces the overall docking process and increases the accuracy of the docking procedure. However, if the ligand does not bind in the region specified by the selected cavity, this option should be disabled. Moreover, randomization of ligand

orientation can be toggled on and off by setting the **Randomize ligand orientation** option. If randomization of orientation is not enabled, the docking search algorithm will be biased towards the orientation of the input ligand to be docked (see Chapter 8 for more details).

Notice: For large ligands with more than 10-15 flexible bonds, 20-50 runs are sometimes needed. Using the MolDock SE search algorithm and the grid-based version of the docking scoring function can reduce the computational load significantly (good results have been reported using this combination and setting the **Number if runs** to 50).

The **General Settings** show the parameters used by the search algorithm. The default values shown are generally suitable for most docking tasks. However, in some cases increasing the **Population size** to 100 can improve the performance of the docking procedure.

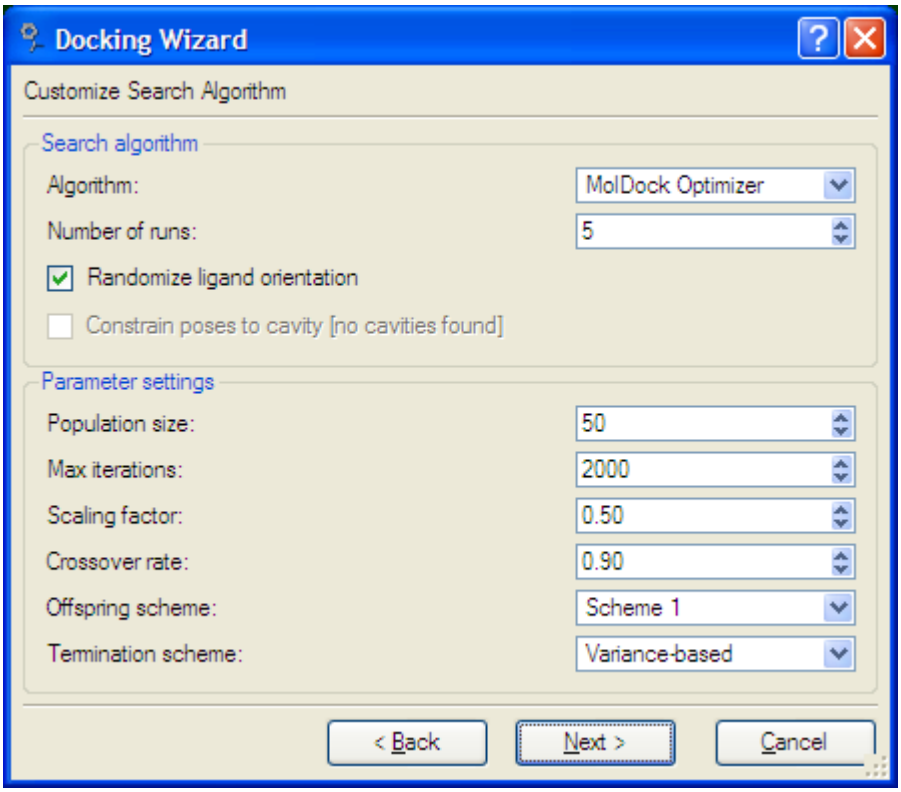


Figure 48: Customizing the search algorithm.

Pose Clustering

Instead of returning only one final pose for each docking run, it is possible to return multiple poses representing different potential binding modes. This can be useful when the best-scoring (i.e. lowest-energy) pose does not represent the native binding mode or when multiple binding modes exists.

Here, clustering can be used to reduce the number of poses found during the

docking run and only the most promising ones will be reported.

Several options are available:

- Limit the number of poses reported (**Max number of poses returned**).
- Only report poses with energies less than a user-defined threshold (**Enable energy threshold**).
- Cluster poses using the specified RMSD threshold (**Cluster similar poses**). Poses found during the docking run will be clustered (put into bins) using the RMSD criteria. See Appendix IV: Clustering Algorithm for a detailed description of the clustering algorithm used. Only the lowest-energy representative from each cluster will be returned when the docking run is completed. Increasing the RMSD threshold will increase the diversity (with respect to RMSD) of the poses returned.
- The **Ignore similar poses** option is used to avoid reporting to similar poses when conducting multiple runs (docking the same ligand). All poses returned from the runs will be clustered and similar poses are removed (keeping the best-scoring one). Depending on the RMSD threshold specified, more or less diverse poses (combined for all the runs) will be reported.

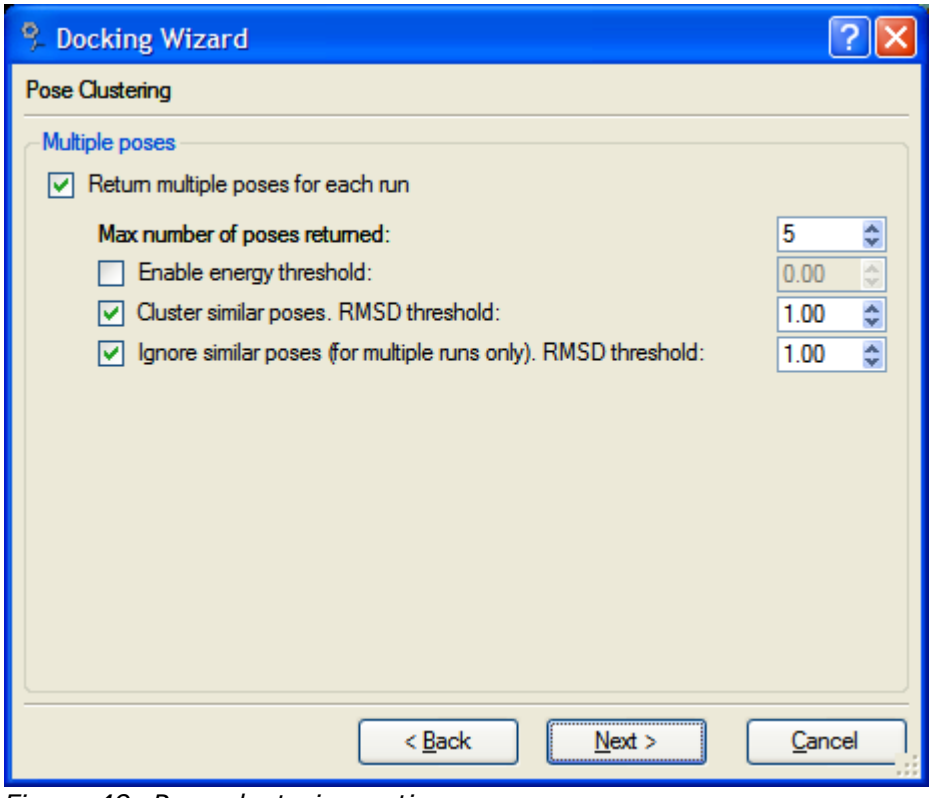


Figure 49: Pose clustering options.

Notice: The actual number of poses returned may be lower than the maximum number of poses specified in **Max number of poses returned**. For instance,

the energy- or clustering-threshold options can reduce the number of poses returned (if poses have higher energies or are too similar). However, the overall best-scoring pose will always be returned.

Errors and Warnings

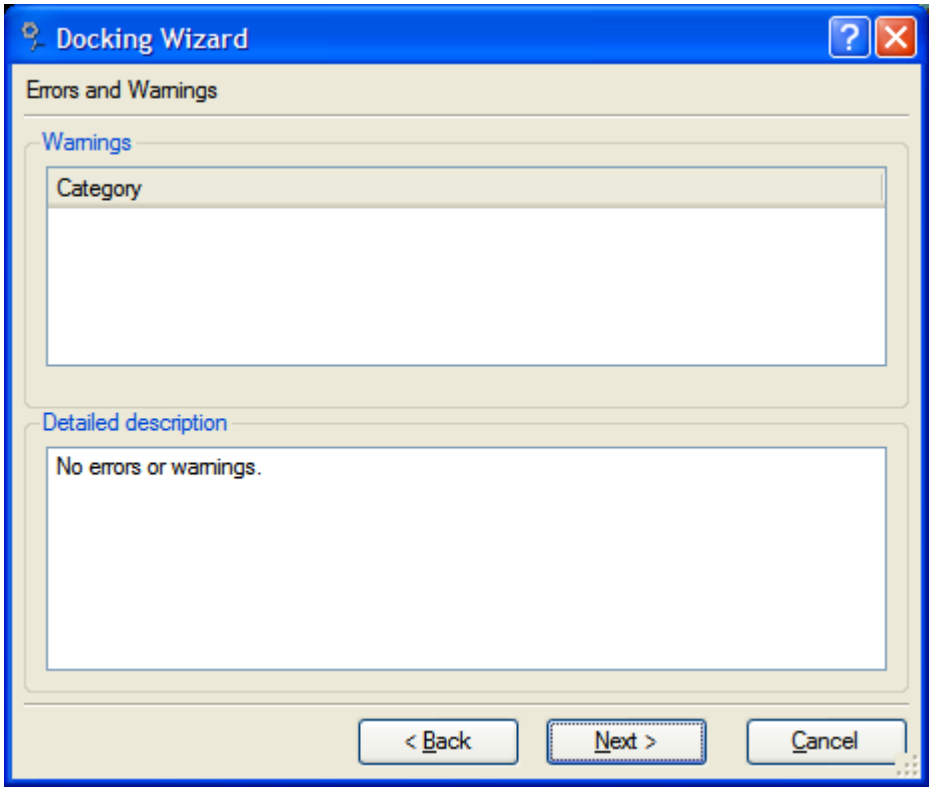


Figure 50: Any warnings or errors are shown on the last page in the wizard.

The **Docking Wizard** reports errors and warnings found, such as non-bonding atoms in molecules, steric clashes between atoms, unsupported residues, missing hydrogens in proteins, etc. A detailed description of each warning and error is shown at the bottom of the **Errors and Warnings** tab (see Figure 50).

Setup Docking Execution

In the final tab (see Figure 53), two choices are available for executing the docking simulation. **Run docking in separate process** is the default choice, which creates a MVD script that is executed in an external process (Chapter 10 describes the MVD Scripting Interface in more details). A copy of the current workspace is used, so the user can continue working with the current workspace without interfering with the docking simulation (e.g. add/remove molecules, change preparation, etc.). The second choice **Create a docking script job, but do not run it now** creates a docking script using the currently selected parameter settings. The generated script is saved in the directory specified in the **Output directory** (see below) and can be used to start up the

docking simulation on other computers.

The **Output directory** specifies where the docking data (MVD script file, MVD script log file, docking results file, and found poses) will be stored. The MVD script file (`script.mvdscript`) contains the scripting commands automatically generated to perform the docking simulation. The MVD script log file (`ScriptLog_timestamp.txt`) is a time-stamped log file containing log information generated by the script interpreter. Details about the poses returned after the docking simulation (e.g. docking score, affinity, specific energy terms, and pose SDF filename) are included in a `mvdresults` file (`DockingResults.mvdresults`). The `mvdresults` file is used by the **Pose Organizer** to show detailed information about the poses and to dynamically load the molecular structure of the poses (see Section 6.1 for more details). Finally, each pose is stored in SDF format. These files are used by the **Pose Organize**r to show the 3D conformations of the poses in the **Visualization Window**.

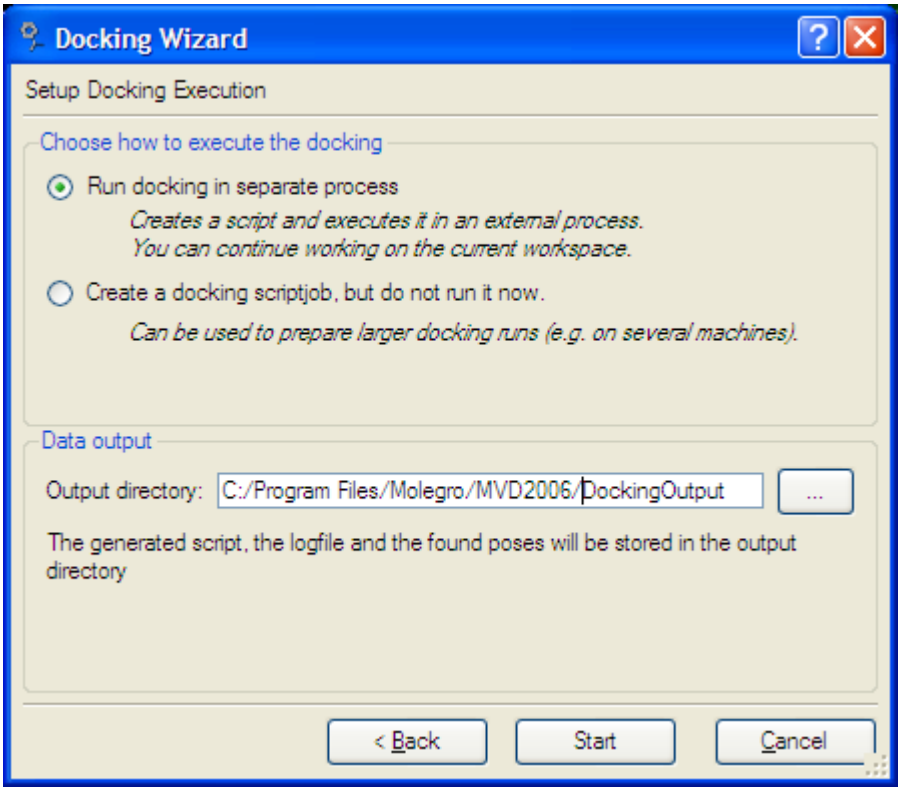

Docking Wizard
Setup Docking Execution
Choose how to execute the docking
Run docking in separate process
Creates a script and executes it in an external process.
You can continue working on the current workspace.
Create a docking scriptjob, but do not run it now.
Can be used to prepare larger docking runs (e.g. on several machines).
Data output
Output directory:
C:/Program Files/Molegro/MVD2006/DockingOutput
...
The generated script, the logfile and the found poses will be stored in the output directory
< Back
Start
Cancel

Figure 51: Setup docking execution.

Finally, when the **Start** button is pressed, the docking run will start and the **Molegro Virtual Docker Batchjob** dialog will pop up - showing the current status and progress of the docking (see Figures 52 and 57).

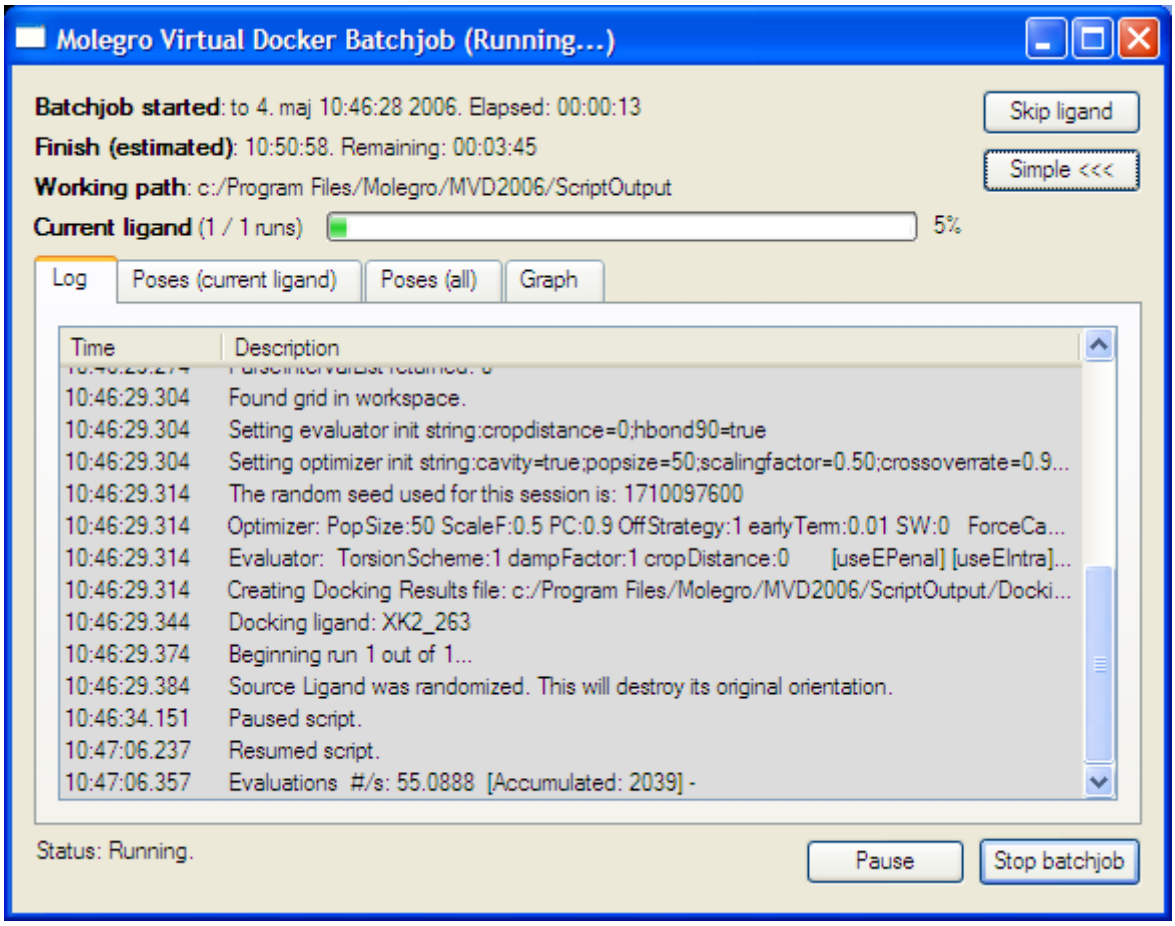

Molegro Virtual Docker Batchjob (Running...)
Batchjob started: to 4. maj 10:46:28 2006. Elapsed: 00:00:13
Finish (estimated): 10:50:58. Remaining: 00:03:45
Working path: c:/Program Files/Molegro/MVD2006/ScriptOutput
Current ligand (1 / 1 runs)
5%
Skip ligand
Simple <<<
Log
Poses (current ligand)
Poses (all)
Graph
Time
Description
10:46:29.304 Found grid in workspace.
10:46:29.304 Setting evaluator init string:cropdistance=0;hbond90=true
10:46:29.304 Setting optimizer init string:cavity=true;popsize=50;scalingfactor=0.50;crossoverrate=0.9...
10:46:29.314 The random seed used for this session is: 1710097600
10:46:29.314 Optimizer: PopSize:50 ScaleF:0.5 PC:0.9 OffStrategy:1 earlyTerm:0.01 SW:0 ForceCa...
10:46:29.314 Evaluator: TorsionScheme:1 dampFactor:1 cropDistance:0 [useEPenal] [useEIntra]...
10:46:29.314 Creating Docking Results file: c:/Program Files/Molegro/MVD2006/ScriptOutput/Docki...
10:46:29.344 Docking ligand: XK2_263
10:46:29.374 Beginning run 1 out of 1...
10:46:29.384 Source Ligand was randomized. This will destroy its original orientation.
10:46:34.151 Paused script.
10:47:06.237 Resumed script.
10:47:06.357 Evaluations #/s: 55.0888 [Accumulated: 2039] -
Status: Running.
Pause
Stop batchjob

Figure 52: Docking Progress dialog

The **Graph** tab shows the convergence of the population of candidate solutions. The blue graph shows the energy of the best pose and the red graph shows the mean energy of the entire population of candidate solutions (see Appendix II: Docking Search Algorithm for more details about the docking simulation and the population terminology).

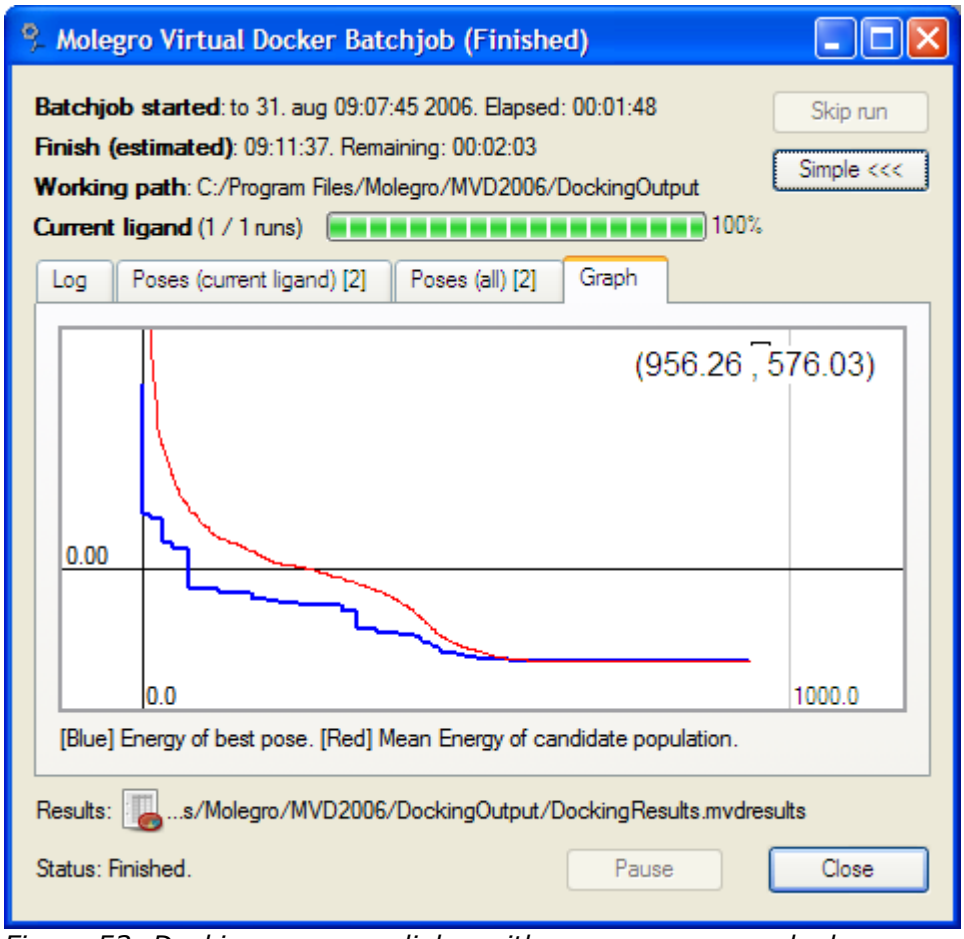


Figure 53: Docking progress dialog with convergence graph shown.

6 Analyzing the Docking Results

6.1 Pose Organizer

The **Pose Organizer** is used to inspect poses found (see Figure 54). It allows you to browse the list of current poses, to see detailed information about specific energy contributions, to visualize hydrogen bonds, electrostatic interactions, and to calculate ranking scores and estimate binding affinity energies.

The **Pose Organizer** can be invoked in several ways. It is automatically displayed after a docking result file (with `mvdresults` file extension) has been imported to MVD by dragging-and dropping the file into MVD, using **File** | **Import Docking Results (*.mvdresults)...**, or by dragging and dropping the DockingResults icon (located in the lower left corner of the **Molegro Virtual Docker Batchjob** dialog) onto the MVD application.

Otherwise it can be invoked by using the context menu on the **Poses** category in the **Workspace Explorer** or using **Docking** | **Pose Organizer** if poses are present in the **Workspace Explorer**.

When the **Pose Organizer** is invoked it displays a list of poses parsed from the `mvdresults` file (or poses currently in the workspace). The table in the middle of the dialog window shows various columns with information about different energy contributions and other data for each pose. The columns can be changed under the **Settings** tab pane. A panel in the bottom of the dialog (**Sorting Criteria**) allows the user to sort the table by up to three different criteria.

By default the table in the middle supports multiple selection, i.e. more than one pose can be highlighted. Only highlighted poses will be visible in the 3D

window. This setting is useful for quick comparison of different poses.

This default behavior can be changed by selecting **Dynamic update (notice: disables multiple poses selection)**. In this mode only one pose is shown at a time. In return it offers the possibility to visualize different interactions for the current selected pose (e.g. hydrogen bonds).

When inspecting poses obtained from different ligands, the **Only show top ...** option can be used focus on the most promising poses for each ligand. The selection of the top poses are based on the currently chosen **Sorting criteria**.

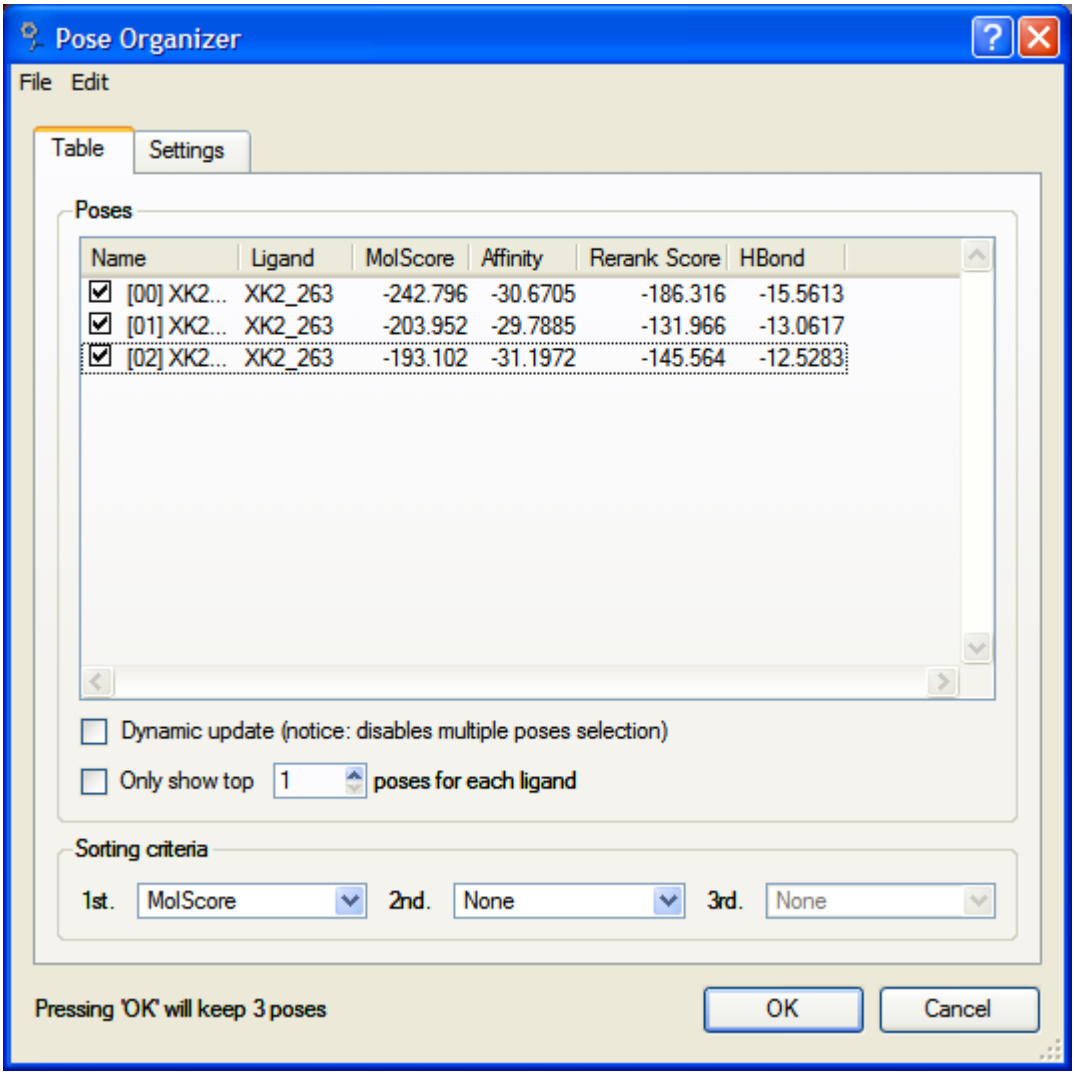

Pose Organizer
File Edit
Table
Settings
Poses
Name Ligand MolScore Affinity Rerank Score HBond
[00] XK2... XK2_263 -242.796 -30.6705 -186.316 -15.5613
[01] XK2... XK2_263 -203.952 -29.7885 -131.966 -13.0617
[02] XK2... XK2_263 -193.102 -31.1972 -145.564 -12.5283
Dynamic update (notice: disables multiple poses selection)
Only show top 1 poses for each ligand
Sorting criteria
1st. MolScore 2nd. None 3rd. None
Pressing 'OK' will keep 3 poses
OK
Cancel

Figure 54: The Pose Organizer dialog.

Notice: A detailed energy analysis is available by right-clicking poses in the table and selecting **Show Detailed Energy Analysis**. Additional options are available in the context menu allowing the user to select, remove, and export poses. These options are also available from the **File** and **Edit** menus located in the **Pose Organizer** dialog.

The **Settings** Tab Pane of the **Pose Organizer** can be used to customize the **Pose Organizer** (see Figure 55).

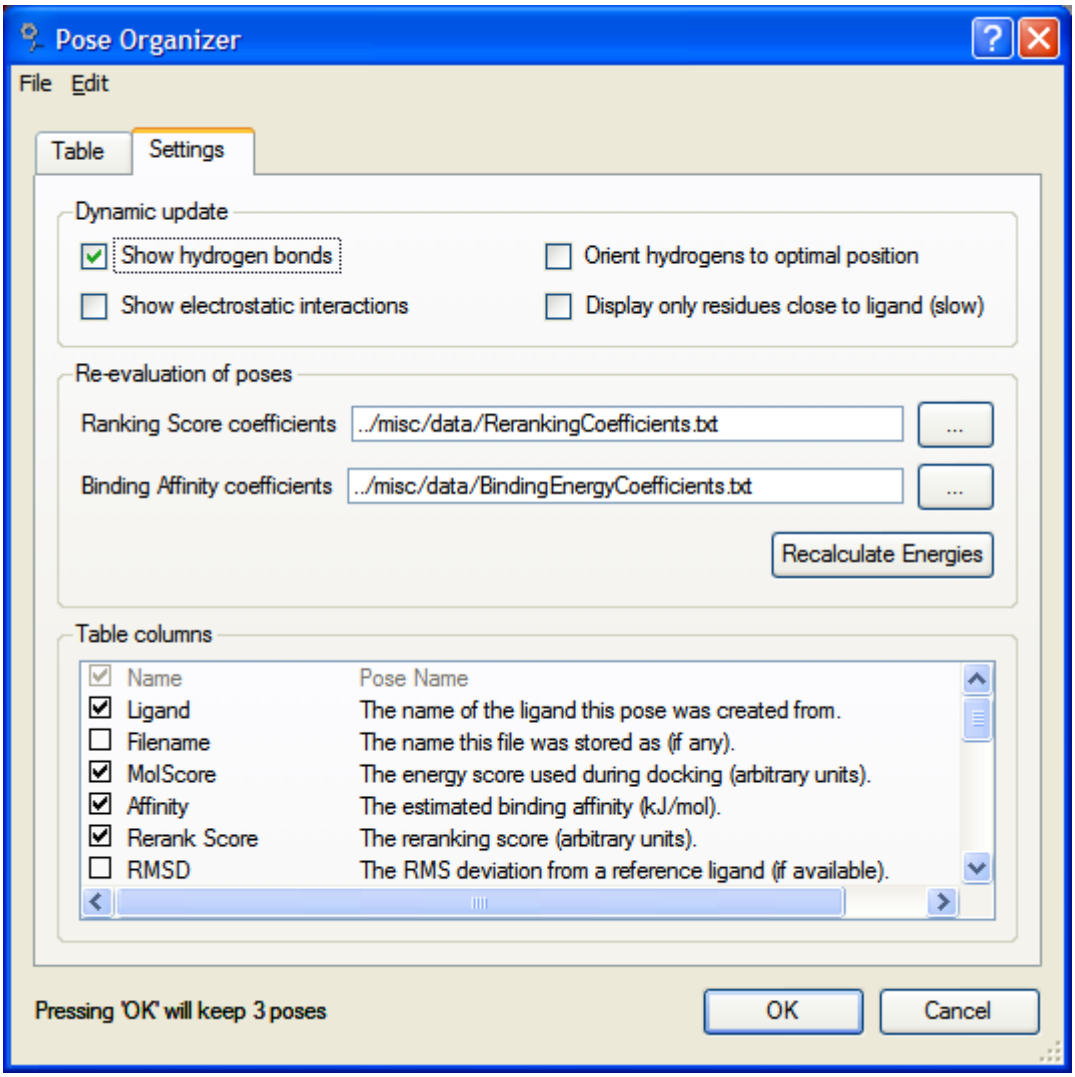


Figure 55: Pose Organizer settings.

The Dynamic Update Panel

The top panel (**Dynamic update**) chooses how the **Pose Organizer** behaves when single pose selection (**Dynamic update**) is enabled. It allows you to visualize hydrogen bonds, electrostatic interactions, orient hydrogens in the protein and ligand to their optimal position, and dynamically show residues close to the chosen pose. The **Orient hydrogens to optimal position** option is useful when inspecting poses as this makes it easier to see if the hydrogen bond is optimal.

The Re-Evaluation of Poses Panel

The middle panel allows for recalculation of scoring functions including MolDock

Score, Binding Affinity score, and re-ranking score. These scoring function values are already calculated if the poses are imported from a `mvdresults` file. Pressing the **Recalculate Energies** button will recalculate the scores for each of the three measures (using the coefficients specified in the files for the binding affinity and re-ranking scores).

The ranking score function is computationally more expensive than the scoring function used during the docking simulation but it is generally better than the docking score function at determining the best pose among several poses originating from the same ligand. The default reranking coefficients are listed in the file: `\Misc\Data\RerankingCoefficients.txt`

The binding affinity is believed to be the best choice when trying to identify the best binder to a given target between a set of different ligands.

The binding affinity of a given pose is given by: $E_{binding}$ = -5.68*pKi and is measured in kJ/mol (the numerical factor corresponds to a temperature of 297K). The pKi is predicted using a combination of energy terms and molecular descriptors and then converted to a binding affinity estimate using the equation above. The coefficients for the binding affinity terms were derived using multiple linear regression. The pKi estimator has been calibrated using a data set of more than 200 complexes with known binding affinities. The correlation coefficient was 0.60 when doing 10-fold cross validation.

In general, estimating the binding affinity is a difficult task. Although the accuracy of our binding affinity estimation is on par with other approaches, our current implementation is considered a work in progress and we expect to improve it further.

The Table Columns Panel

The bottom panel (**Table columns**) determines which columns (descriptors) that are shown in the table on the first tab. Table 1 describes the descriptors that are available.

Column Name	Description
Name	Pose name
Ligand	The name of the ligand this pose was created from
Filename	The name this file was stored as (if any)
MolDockScore	The energy score used during docking (arbitrary units)
Affinity	The estimated binding affinity (kJ/mol)
Rerank Score	The reranking score (arbitrary units)
RMSD	The RMS deviation from a reference ligand (if available)
Interaction	The total interaction energy between the pose and the target molecule(s)
Cofactor	The interaction energy between the pose and the cofactors
Protein	The interaction energy between the pose and the protein
Water	The interaction energy between the pose and the water molecules
Internal	The internal energy of the pose
Solvation	The energy calculated from the implicit solvation model
Protein VdW	Protein steric interaction energy from an alternative 12-6 VdW potential
Torsions	The number of (chosen) rotatable bonds in the pose
Soft Constraints	The energy contributions from soft constraints
Electro	The electrostatic energy in the model
HBond	Hydrogen bonding energy
Heavy Atoms	Number of ligand heavy atoms
LE1	Ligand Efficiency 1: MolScore divided by Heavy Atoms count
LE2	Ligand Efficiency 2: Binding Affinity divided by Heavy Atoms count
LE3	Ligand Efficiency 3: Rerank Score divided by Heavy Atoms count

Table 1: Column names available in the Pose Organizer dialog.

6.2 Saving Molecules and Solutions Found

Saving Workspace

After importing and preparing molecules, all information can be saved in a MVD Workspace (MVDML) file, which contains all relevant information (position of atoms, charges, hybridization, bond orders, ligand flexibility, ...). To save a workspace, select **File** | **Save Workspace As...**. Alternatively, use the keyboard shortcut **Ctrl-S**.

Notice: Visualization objects (surfaces, labels, cavities, interactions, ...), state information (active and reference ligand, structural water enabled/disabled, ...) are not saved in MVDML files.

Exporting Molecules

The **Export Molecules** dialog can be used to export all (or a selection of) the molecules available in the workspace (see Figure 56).

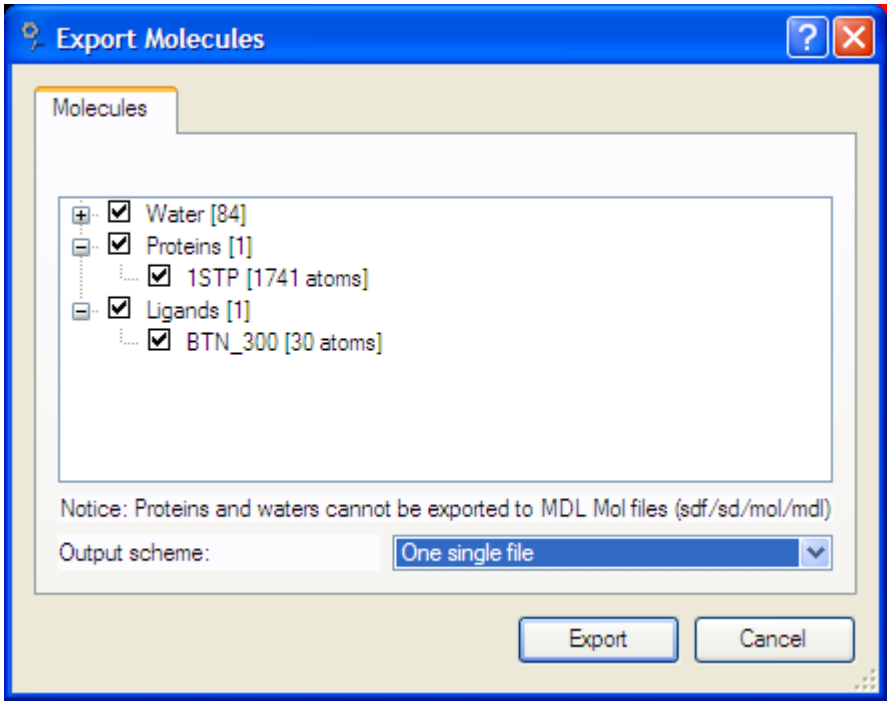


Figure 56: Export Molecules dialog: Select which molecules to export.

To export molecules, select **File** | **Export Molecules...** or **Export Molecules...** from the **Workspace** context menu in the **Workspace Explorer** (also available for proteins, ligands, cofactors, and poses).

Notice: Proteins and water molecules cannot be exported to MDL Mol files.

Exporting Poses Found

To save the poses obtained from the docking runs, either use the **Export Molecules** dialog (described above) or save the poses from the **Pose Organizer** dialog.

6.3 Pose Modifier

It is possible manually to modify a ligand (or a pose found) by right-clicking the molecule in the **Workspace Explorer** and selecting **Modify Pose** (see Figure 58). When invoking the **Pose Modifier**, a new pose is created.

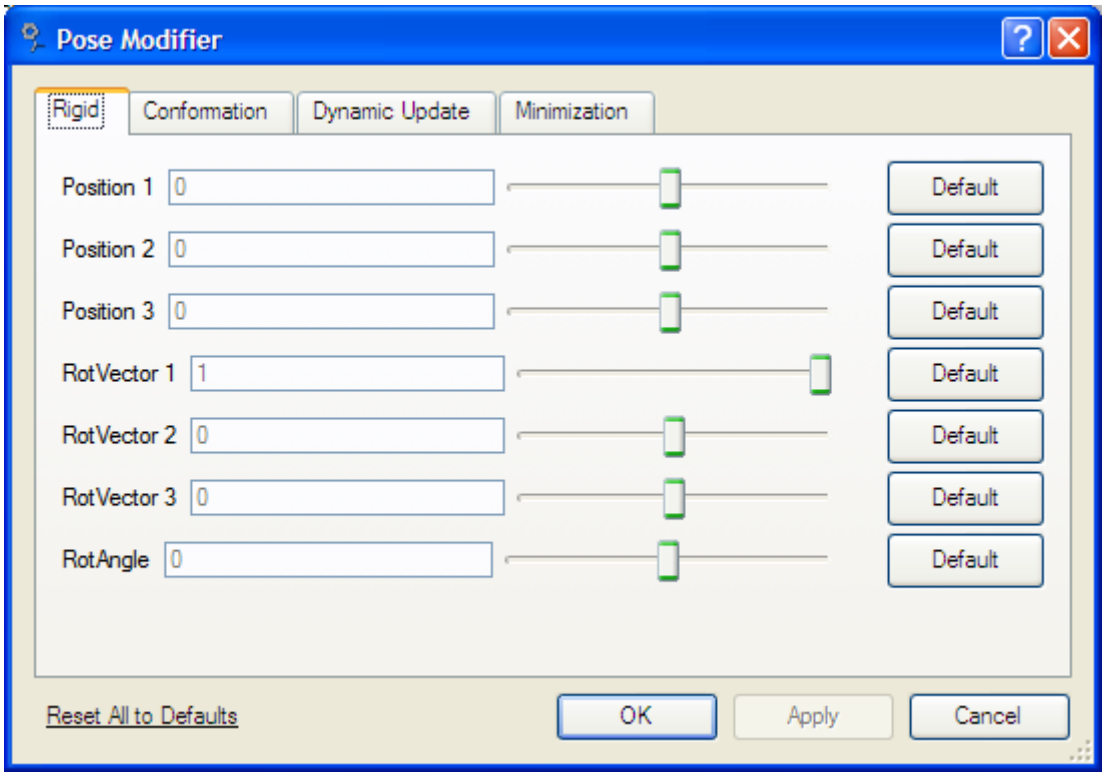


Figure 57: Pose Modifier dialog.

Notice: It is not possible to directly modify poses after the workspace has been saved and reloaded. However, ligands can be modified any time. To modify poses saved, these can be converted to ligands and modified afterwards (which will result in a new modified pose). Different interactions can also be visualized on-the-fly (**Dynamic Update** tab).

6.4 RMSD Matrix

The **RMSD Matrix** dialog can be used to quickly inspect deviations between molecules in the workspace. In addition to the standard measure **Pairwise Atom-Atom RMSD (by ID)**, two variants **Pairwise Atom-Atom RMSD (checking all automorphisms)** and **Pairwise Atom-Atom RMSD (by**

nearest unmatched neighbour) of the RMSD measure tries to take intrinsic symmetries of the molecule into account when calculating RMSD. The recommended choice is **Pairwise Atom-Atom RMSD (checking all automorphisms)**, which is also used by default.

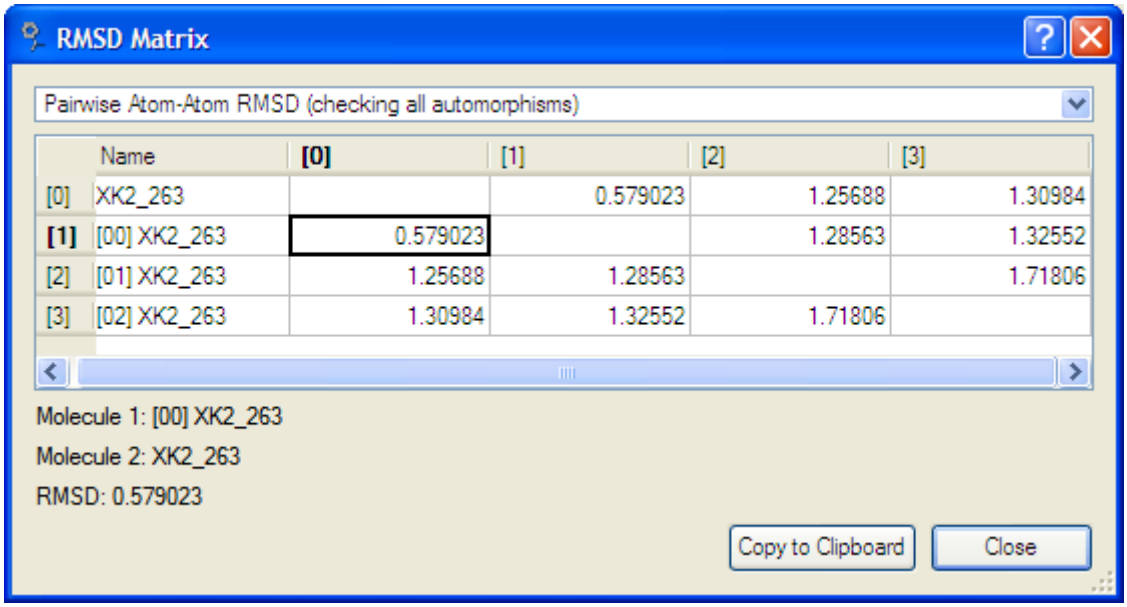


Figure 58: RMSD Matrix dialog.

The dialog can be invoked by choosing **RMSD Matrix** from the **Docking** menu. The **Copy to Clipboard button** can be used to copy the table to the clipboard for further inspection in an external text editor or spreadsheet.

6.5 Structural Alignment

Simple alignment of molecules is possible. By selecting three atoms in one ligand, and selecting three atoms in another ligand, a new context menu appears when clicking on an atom in one of the molecules - **Align...**. This will align the molecules. The atoms are aligned in the same order as they are selected, that is, the first selected atom in ligand 1 is aligned to the first selected atom in ligand 2 etc. Therefore, it is important to ensure that the selection order is correct and that no other atoms are selected. Alignments can be undone (click the undo button in the tool bar).

Notice: Only alignments with three selected atoms in each molecule are possible.

7 Customizing Molegro Virtual Docker

7.1 General Preferences

Molegro Virtual Docker can be customized using the **Preferences** dialog, which can be invoked from the **Window** or **Edit** menus or by pressing **F4**. Preference settings are categorized in **General**, **Graphics**, and **Debug** tabs.

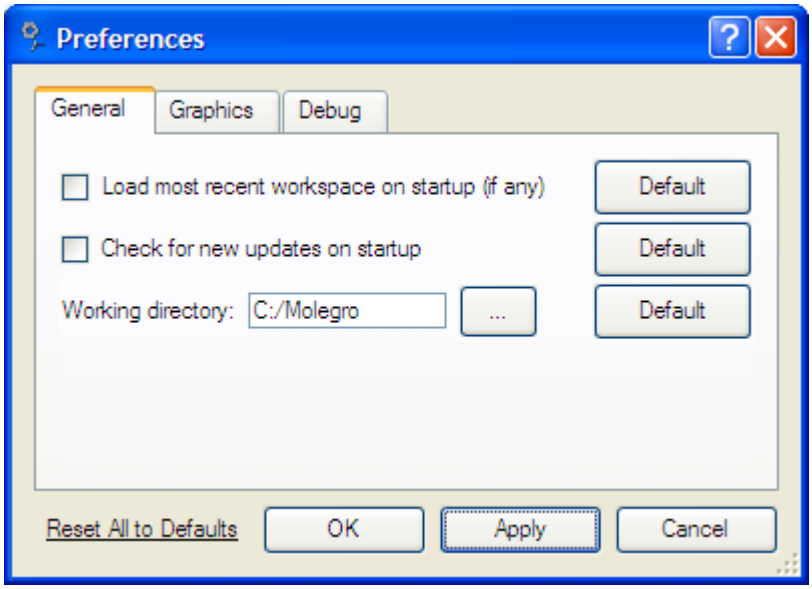


Figure 59: First tab of the preferences dialog.

In the **General** tab (see Figure 59), the following settings are available:

- The **Load most recent workspace on startup (if any)** option toggles automatic import of the last used workspace.

- The **Check for new updates on startup** option enables MVD to automatically check for new updates during startup.
- The **Working directory** setting is used to set the current **Working directory**, which is the root path for file related operators (e.g. when loading and saving molecular structure files and log files).

The **Graphics** tab (see Figure 60) contains settings related to the **Visualization Window**:

- The **Show pivot point (rotational center)** option toggles the visibility of the pivot point (small grayish ball).
- The **Show root atom** option toggles the visibility of the currently chosen root atom for each of the ligands in the workspace (see 'Set root atom' in Section 4.3 for more info).
- The **Fade 3D labels when in background** option toggles fading of labels in the **Visualization Window.**
- The overall rendering quality can be specified using the **Quality** option. Modern computers with dedicated 3D hardware should be able to run at highest quality even when rendering relatively large molecules. It is easy to test new quality settings by selecting the level of quality and pressing the **Apply** button.

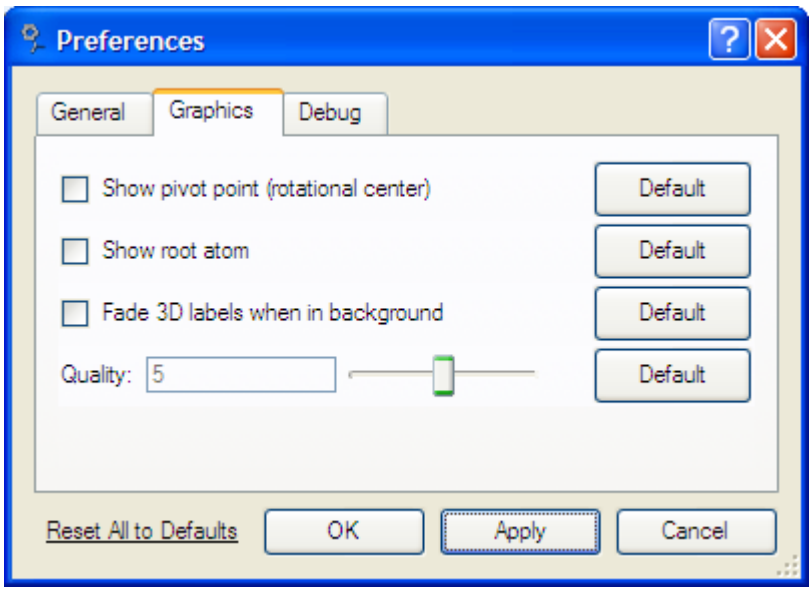


Figure 60: The graphics tab of the Preferences dialog.

The final settings tab, **Debug**, contains various options that can be used to provide additional debug information.

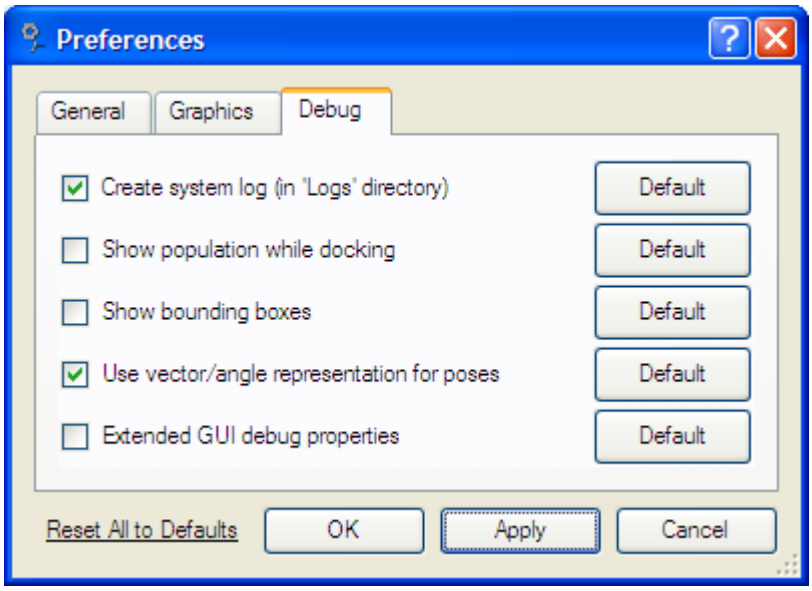


Figure 61: The debug tab of the Preferences dialog.

For standard uses, it is not recommended to change the default settings of any of these options. The **Create system log (in 'Logs' directory)** option is used to toggle whether a system log is created for each execution of MVD. The system log contains information of user actions conducted and is used to track potential bugs and performance problems. The log files are stored within the `Logs` directory located in the same directory as the `mvd` executable file.

Notice: If you encounter problems with MVD please email the log file created before the crash to: bugs@molegro.com

7.2 Command Line Parameters

Currently, the following command line parameters are available:

```
<filename>
-nogui
-interactive
-currentPath
```

The `<filename>` parameter can be used to import molecular files during MVD startup. If more than one file is listed (separated by spaces), each file will be imported.

Example: `/Molegro/MVD2006/bin/mvd 1stp.pdb`

If the filename has `mvdscript` as file extension (e.g. `mydocking.mvdscript`), a script parsing progress dialog will be invoked and the script will be parsed and interpreted.

The `-nogui` parameter can be used to run the script job without invoking the progress dialog.

Example: `/Molegro/MVD2006/bin/mvd mydocking.mvdscript -nogui`

Using the `-interactive` parameter, MVD can be started in interactive mode which is used to allow scripting languages (e.g. Python) to interact with MVD and control the docking process. See Chapter 10 for more details.

Finally, the `-currentPath` parameter can be used to override the working directory specified in the general preference settings with the current path. This is particularly useful when running MVD from different working directories (using a terminal window) or when using a script to start up MVD.

Example: `/Molegro/MVD2006/bin/mvd -currentPath`

7.3 Changing Ranking Score Coefficients

The energy terms and their weights (coefficients) used in the ranking scoring function can be altered by modifying the `RerankingCoefficients.txt` file located in the `/Misc/Data/` directory (located within the main directory of MVD).

Notice: Changing these coefficients and disabling/enabling energy terms will alter the performance of the ranking score used in the **Pose Organizer** dialog and may result in much worse performance. Remember to backup the original file before modifying the coefficients.

8 Obtaining the Best Docking Results

This section takes a closer look at the most important aspects regarding preparation, docking, and post-analysis that can be decisive for whether docking with Molegro Virtual Docker will be successful or not. By taking the following suggestions into account, we hope that common pitfalls can be avoided.

8.1 Preparation

- General issues: It is recommended to remove unwanted material such as proteins, ligands, cofactors, and water molecules if they are not needed in the actual docking simulation.
- Validation: The automatic preparation of molecules might fail in some cases. It is therefore advisable to manually inspect the molecules (in particular ligands) and check bond orders, hybridization states, and if hydrogens are correctly assigned.
- Protonation: If the protein is expected to have unusual protonation states near the binding site, be sure to set them using the **Protonation Wizard**.
- Ligand flexibility: By default, all torsions in the ligand that can be flexible are set flexible during the docking simulation. The complexity of the docking search can be significantly reduced, if the number of torsions that are set flexible during the docking run is lowered. Bonds can be set rigid during docking using the context menu (right-click on the bond and select **Set Flexibility** | **Rigid while docking**).
- Cavity detection: Before starting the docking run, all potential binding sites (active sites) should be identified using the **Detect Cavities** dialog.

The default settings listed in the wizard are generally applicable. However, for large proteins or proteins having a lot of cavities, it is sometimes necessary to increase the number of cavities reported (**Max number of cavities**). Also remember to set the binding site **Origin** (in the **Docking Wizard**) to the specific cavity being investigated.

- Domain knowledge: The success of the docking run can be significantly improved if any domain knowledge is available. For instance, knowledge about preferred binding mode or ligand conformation can be used to set constraints or reduce the search space covered (e.g. Constraints and binding site settings in the **Docking Wizard**).
- In some cases, docking performance can be improved by selecting another ligand root atom (right-click on ligand atom and select **Set as Root Atom**. The current root atom can be visually identified if visualization of root-atoms is enabled (see Section 7.1). The root atom is used as root in the torsion tree that is constructed when docking flexible ligands. Docking performance may be improved by setting the root atom in a region of the ligand that is suspected to contribute significantly to the docking energy.

8.2 Docking

- Size of search space: The size and location of the volume that the docking search algorithm will sample is defined by the **Binding site** settings in the **Docking Wizard**. Before starting the docking run, potential cavities should be identified (see Section 5.1). Found cavities can be used to specify the origin of the search space (in the **Docking Wizard**) and constrain candidate solutions to the region covered by the cavity (by enabling the **Constrain poses to cavity** option in the **Docking Wizard**).

  Notice: It is important to select a search space **Radius** that allows the ligand to be positioned within the search space region (typically between 15 and 20 Å). However, the **Radius** should be set as small as possible to make the docking search efficient. Likewise, the **Origin** (center) of the search space region can be manually adjusted to focus the sampling of candidate solutions to a specific region. This is particularly important if the cavity volume is much bigger than the ligand (for large cavities, focusing on one specific part of the cavity will significantly increase the docking accuracy).
- Search parameters: The default settings for the docking search algorithm are generally applicable. However, in some cases (e.g. for ligands with more than 15 torsions) it can be advantageous to increase the **Population size** to 100 individuals or more.

- Multiple runs: Because of the stochastic nature of the docking search algorithm, it is recommended to make multiple runs for each ligand-protein setup. Typically, about 5-10 runs are needed to ensure convergence to the lowest-energy solution. For large ligands with more than 10-15 flexible bonds, 20-50 runs are sometimes needed. When conducting more than one docking run, **Randomize ligand orientation** should be enabled (see **Biased Orientation** bullet below). Additionally, it is recommended to ignore similar poses returned (see Section 5.3) to lower the number of similar poses reported when taking all docking runs into account.
- Multiple poses: It is advisable to return multiple poses for each docking run (typically between 3 and 10) and rerank the poses found afterwards (see **Ranking poses** bullet below).
- Check warnings: The last tab in the **Docking Wizard** highlights potential warnings and errors. It is important to inspect the warning messages and see if further actions are needed. Otherwise, the docking run might be unsuccessful.
- Biased and randomized orientation: Randomization of ligand orientation can be toggled on and off by setting the **Randomize ligand orientation** in the **Docking Wizard**. If randomization of orientation is not enabled, the docking search algorithm will be biased towards the orientation of the input ligand to be docked.

  Notice: Using the *biased* orientation can be advantageous if the orientation of the ligand to be docked is known beforehand or if the orientation is somewhat similar to the input structure. For instance, if another inhibitor is known, it can be used to align the ligand to the inhibitor before the docking run is started (see Section 6.5 for more details). In these cases, a biased orientation will allow the docking search algorithm to focus more on that specific orientation (or nearby orientations) leading to more accurate docking results.

8.3 Post-analysis

- Ranking poses: The most promising poses returned when the docking run terminates can be further analyzed in the **Pose Organizer**. Ideally, the highest-scoring pose should represent the best-found binding mode. However, this is not always the case. A useful feature is to evaluate the poses using either the **Ranking Score** or the **Binding Affinity** estimate. The **Ranking Score** makes use of a more advanced scoring scheme than the docking scoring function used during the docking run. Using the Ranking Score will often increase the accuracy of the ranked order of the poses.

9 Help

9.1 Online Help

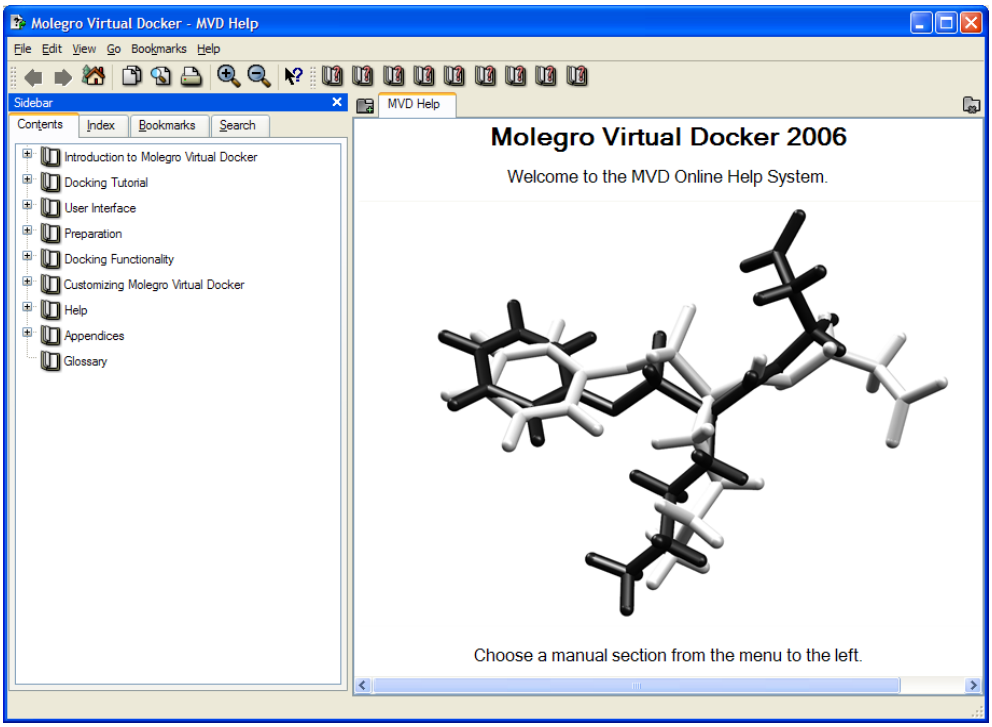


Figure 62: Screenshot from the online help system.

The documentation for Molegro Virtual Docker is available both as indexed online help (HTML formatted) and as a PDF file. In order to invoke the online help, choose **Help** | **MVD Help** from the menu bar.

In the sidebar you can browse the different chapters in the **Contents** tab. It is also possible to perform a full-text search by choosing the **Search** tab from the sidebar.

9.2 Tip of the Day

A 'Tip-of-the-Day' dialog (see Figure 63) providing useful tips on how to use Molegro Virtual Docker is available.

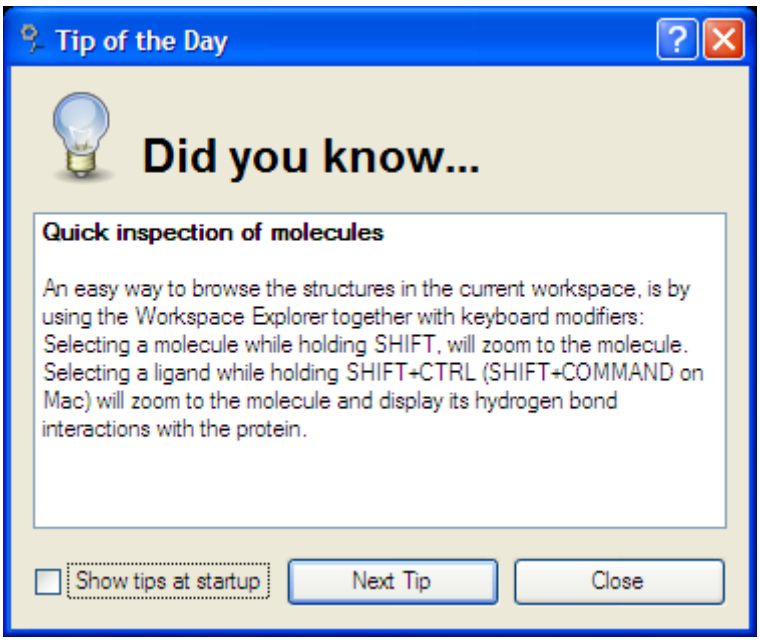

Tip of the Day
Did you know...
Quick inspection of molecules
An easy way to browse the structures in the current workspace, is by using the Workspace Explorer together with keyboard modifiers: Selecting a molecule while holding SHIFT, will zoom to the molecule. Selecting a ligand while holding SHIFT+CTRL (SHIFT+COMMAND on Mac) will zoom to the molecule and display its hydrogen bond interactions with the protein.
Show tips at startup
Next Tip
Close

Figure 63: 'Tip of the Day' dialog.

The dialog can be manually invoked from the **Help** menu or automatically shown on startup. The automatic startup setting can be toggled in the dialog or from the general **Preferences** dialog.

9.3 The Molegro Website

The Molegro website also offers certain help facilities. Please visit www.molegro.com to see our FAQ's and other information available.

9.4 Technical Support

Technical support is available for commercial licenses (industrial and academic) only. To obtain additional support send an email to support@molegro.com.

10 Script Interface

10.1 Using the Script Interface

The default behavior for docking molecules in Molegro Virtual Docker, is to start the application, load and prepare the molecules, and invoke the **Docking Wizard**.

The **Docking Wizard** guides the user through the different settings for the simulation, and then creates a small script file which contains instructions on how the docking should proceed.

The default behavior for the **Docking Wizard** is then to spawn a script interpreter (in another process – the script interpreter and the main application runs completely separated) and execute the script.

However greater flexibility is possible by writing custom scripts: for instance this makes it possible to dock a number of ligands against several distinct targets. It is also possible to split large docking runs into several scripts and run them on different machines.

Notice: A MVD script job basically runs in a single thread. This means that as such, MVD will not utilize multiple CPU's (or dual-core processors). However by splitting the job into two (or more) jobs and running them concurrently all available CPU's can be utilized.

10.2 Running a Text-file Script

Text-file scripts are ordinary text files saved with the `mvdscript` file extension.

In order to run a text-file script, simply start MVD with the text-file script name as the argument:

Example: `mvd docktest.mvdscript`

This will spawn the Script Progress GUI with information on how the script parsing is progressing:

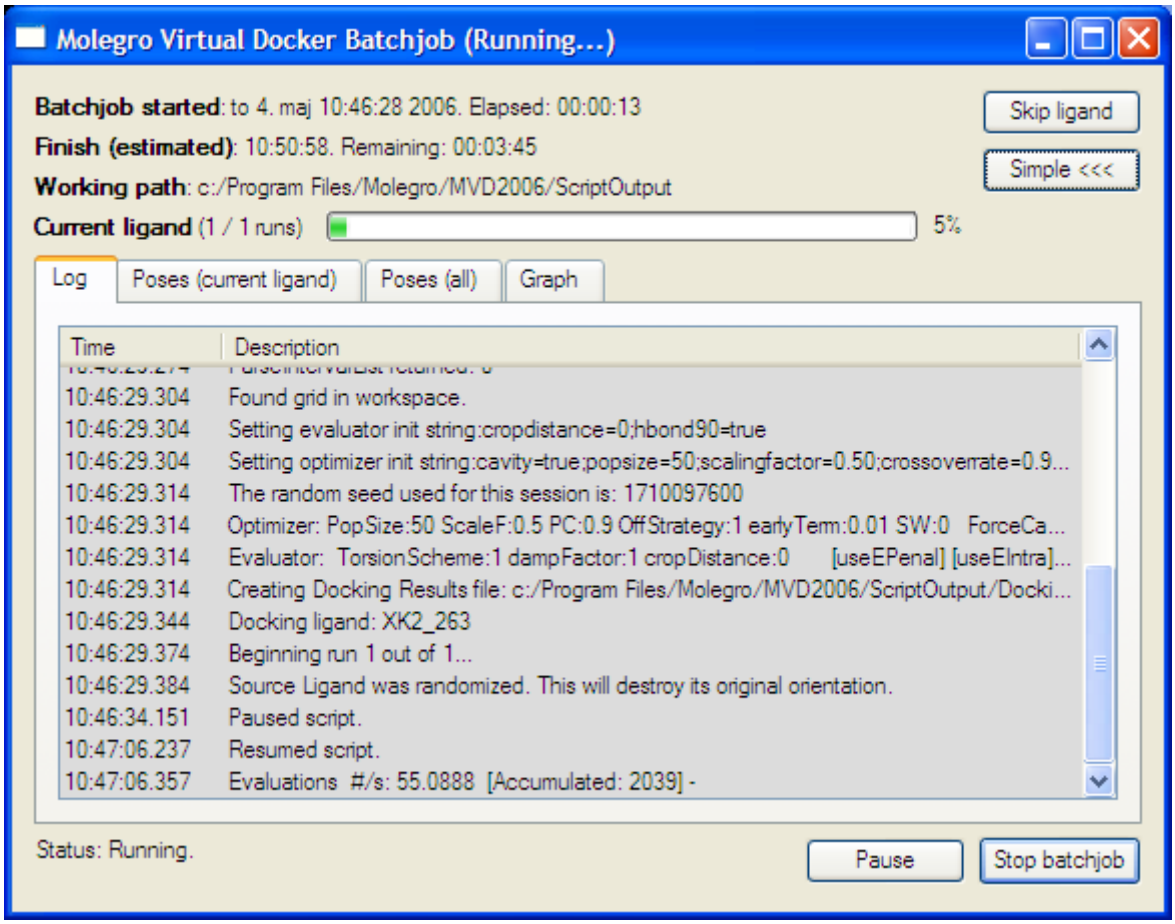

Molegro Virtual Docker Batchjob (Running...)
Batchjob started: to 4. maj 10:46:28 2006. Elapsed: 00:00:13
Finish (estimated): 10:50:58. Remaining: 00:03:45
Working path: c:/Program Files/Molegro/MVD2006/ScriptOutput
Current ligand (1 / 1 runs)
5%
Skip ligand
Simple <<<
Log
Poses (current ligand)
Poses (all)
Graph
Time
Description
10:46:29.304 Found grid in workspace.
10:46:29.304 Setting evaluator init string:cropdistance=0;hbond90=true
10:46:29.304 Setting optimizer init string:cavity=true;popsize=50;scalingfactor=0.50;crossoverrate=0.9...
10:46:29.314 The random seed used for this session is: 1710097600
10:46:29.314 Optimizer: PopSize:50 ScaleF:0.5 PC:0.9 OffStrategy:1 earlyTerm:0.01 SW:0 ForceCa...
10:46:29.314 Evaluator: TorsionScheme:1 dampFactor:1 cropDistance:0 [useEPenal] [useEIntra]...
10:46:29.314 Creating Docking Results file: c:/Program Files/Molegro/MVD2006/ScriptOutput/Docki...
10:46:29.344 Docking ligand: XK2_263
10:46:29.374 Beginning run 1 out of 1...
10:46:29.384 Source Ligand was randomized. This will destroy its original orientation.
10:46:34.151 Paused script.
10:47:06.237 Resumed script.
10:47:06.357 Evaluations #/s: 55.0888 [Accumulated: 2039] -
Status: Running.
Pause
Stop batchjob

Figure 64: Script Progress GUI

Notice: MVD scripts need to have the `.mvdscript` file extension. Otherwise, the script file will not be recognized (and parsed) by MVD.

It is also possible to start a script job with no graphical user interface (without the script parsing progress dialog). This can done by using the `-nogui` command line argument:

Example: `mvd docktest.mvdscript -nogui`

Notice: If you intend to run background jobs on remote Linux/X11 systems, use the `-nogui` argument. Otherwise the system might kill the process when the user logs off (because the X11 server might be shutdown).

10.3 Examples of Common Script Jobs

This section contains some examples of common script jobs. Another useful way of exploring the MVD script syntax is to inspect the script files generated

by the **Docking Wizard**: these files are stored as ordinary MVD script files in the specified directory and can be opened using a standard text editor.

A complete list of commands can be found in Appendix X: Script Commands.

Docking a Single Complex

```
// Init
DOCKSETTINGS maxIterations=1000;runs=1;MaxPoses=5
EVALUATOR cropdistance=0;hbond90=true;water=true
OPTIMIZER cavity=true;popsize=50;crossoverrate=0.9;keepmaxposes=5

// Dock
LOAD 3PTB.MVDML
RMSD ligand[0]
DOCK
```

Docking Multiple Complexes

```
// Init
DOCKSETTINGS maxIterations=1000;runs=1;MaxPoses=5
EVALUATOR cropdistance=0;hbond90=true;water=true
OPTIMIZER cavity=true;popsize=50;crossoverrate=0.9;keepmaxposes=5

// Dock
FOR $MVDML IN 3PTB,1HVR,1LIC,1TMN
   // $MVDML will be replaced by the appropriate value in the loop
   LOAD C:\BENCHMARK\$MVDML.mvdml
   RMSD ligand[0]
   DOCK
   NEW
ENDFOR
```

Splitting Docking Runs Into Multiple Runs

This script can be used to divide the workload between different machines.

```
// Init with appropriate settings first...
DOCKSETTINGS maxIterations=1000;runs=10;MaxPoses=5
EVALUATOR cropdistance=0;hbond90=true;water=true
OPTIMIZER cavity=true;popsize=50;crossoverrate=0.9;keepmaxposes=5
```

```
// For machine 1:
LOAD C:\BENCHMARK\1HVR.mvdml
IMPORT LIGAND[0-99] FROM DB.sdf
DOCK

// For machine 2:
LOAD C:\BENCHMARK\1HVR.mvdml
IMPORT LIGAND[100-199] FROM DB.sdf
DOCK
```

10.4 Running the Script Interface Interactively

MVD can also run in interactive mode.

In this mode the MVD application starts and waits for user input from the command line (i.e. it reads and writes from the standard input and output which can be piped).

To start MVD in interactive mode, use the following syntax:

Example: `mvd -interactive`

The purpose of the interactive mode is to allow scripting languages capable of writing to and from the standard input and output of a program to control the docking process. This can be useful for automating larger docking runs.

When in interactive mode, MVD will send an '[DONE]: <command>' after each command has been interpreted.

10.5 Running the Script Interface From Python

A small Python wrapper is provided in:

`MVD2006/Scripting/Python/MvdWrapper.py`

The wrapper encapsulates the various script commands in a small object, `MVDWrapper`.

The wrapper spawns a new MVD process when the object is instantiated and runs MVD in interactive mode to pass commands to it.

The process can be terminated by calling `exit` on it.

In order to use the wrapper, copy the `MvdWrapper.py` file to the same location as your Python-script (or install it in a globally accessible location) and `import` it at the top of you script.

Notice: The Python wrapper requires Python 2.4 (or above).

The following example is taken from:

`MVD2006/Scripting/PythonWrapper/SimpleDockingTest.py`

```
import os
import MvdWrapper

# create an output dir
outputPath = 'outputData'
complex = '1hvr'

if (not os.path.exists( outputPath )):
       os.mkdir( outputPath )

if os.path.exists( outputPath ) and os.path.isdir( outputPath ):
       print 'Created outputPath: ' + outputPath
else:
       raise IOError, 'could not create path' + outputPath

# Now start the wrapper...
# Remember to change the path to the executable in the line below:
mvd = MvdWrapper.MvdWrapper("C:/Program Files/Molegro/MVD2006/bin/mvdconsole.exe", gui=True)
mvd.info("testing")
mvd.random(123232)                          # set the seed
mvd.cd(outputPath)                          # change to output path
mvd.download(complex, complex + ".pdb")     # download from pdb.org
mvd.importFrom("All", complex + ".pdb")     # import into workspace
mvd.rmsd("ligand[0]")                       # set a ligand as a rmsd reference
mvd.dock("")                                # start the docking
mvd.exit()
```

Notice for Windows Users:

In order to use the Python wrapper you must install the "Python for Windows extensions" which can be downloaded from:

`http://sourceforge.net/project/showfiles.php?group_id=78018`

Notice that you must download the version which targets your specific Python version. Also notice that in order to communicate through pipes with the MVD application be sure to instantiate with a reference to the 'MVDConsole.exe' instead of the standard 'MVD.exe' application. Use:

```
mvd = MvdWrapper.MvdWrapper("C:\Program Files\Molegro\MVD2006\Bin\MVDConsole.exe")
```

instead of: `mvd = MvdWrapper.MvdWrapper("C:\Program Files\Molegro\MVD2006\Bin\MVD.exe")`

11 Appendix I: Docking Scoring Function

The MolDock scoring function (MolDock Score) used by MVD is derived from the PLP scoring functions originally proposed by Gehlhaar et al. [GEHLHAAR 1995,1998] and later extended by Yang et al. [YANG 2004]. The MolDock scoring function further improves these scoring functions with a new hydrogen bonding term and new charge schemes. The docking scoring function, E_{score}, is defined by the following energy terms:

$$E_{score} = E_{inter} + E_{intra}$$

where E_{inter} is the ligand-protein interaction energy:

$$E_{inter} = \sum_{i \in ligand} \sum_{j \in protein} \left[E_{PLP}(r_{ij}) + 332.0 \frac{q_i q_j}{4 r_{ij}^{\ 2}} \right]$$

The summation runs over all heavy atoms in the ligand and all heavy atoms in the protein including any cofactor atoms and water molecule atoms that might be present. The E_{PLP} term is a piecewise linear potential described below. The second term describes the electrostatic interactions between charged atoms. It is a Coulomb potential with a distance-dependent dielectric constant given by: $D(r) = 4r$. The numerical value of 332.0 fixes the units of the electrostatic energy to kilocalories per mole. To ensure that no energy contribution can be higher than the clash penalty the electrostatic energy is cut-off at the level corresponding to a distance of 2.0 Å for distances less than 2.0 Å. Notice that although the electrostatic energy contribution has the theoretically predicted

magnitude, the other energy terms are empirically motivated and the total energy does not necessarily correlate with the true binding affinity. The charges are set according to the scheme listed in Table 2. Metal ions are assigned a charge of +1 (e.g. *Na*) or +2 (e.g. *Zn*, *Ca*, *Fe*).

charge	ligand atoms	protein atoms
0.5	N atoms in $-C(NH_2)_2$	His (ND1/NE2) Arg (NH1/NH2)
1.0	N atoms in $-N(CH_3)_2$, $-(NH_3)$	Lys (N)
-0.5	O atoms in –COO, $-SO_4$, $-PO_2$, $-PO_2-$	Asp (OD1/OD2) Glu (OE1/OE2)
-0.66	O atoms in $-PO_3$	
-0.33	O atoms in $-SO_3$	
-1.0	N atoms in $-SO_2NH$	

Table 2: Charge templates.

E_{PLP} is a "piecewise linear potential" using two different sets of parameters: One set for approximating the steric (Van der Waals) term between atoms, and another stronger potential for hydrogen bonds. The linear potential is defined by the following functional form:

$$E_{PLP}(0) = A_0,\ E_{PLP}(R_1) = 0,\ E_{PLP}(R_2) = E_{PLP}(R_3) = A_1,\ E_{PLP}(r) = 0 \text{ for } r \geq R_4$$

and is linearly interpolated between these values. The parameters used here (see Table 3) were adopted from GEMDOCK [YANG 2004].

	A_0	A_1	R_1	R_2	R_3	R_4
hydrogen bond	20.0	-2.5	2.3	2.6	3.1	3.6
steric	20.0	-0.4	3.3	3.6	4.5	6.0

Table 3: PLP parameters.

A bond is considered a hydrogen bond if one of the atoms can *donate* a hydrogen atom and the other atom can *accept* it. The atom types are assigned according to the scheme shown in Table 4.

type	atoms
acceptor	N and O (with no Hs attached)
donor	N and S (with one or more Hs attached)
both	O (with one H attached) or O in structural water
nonpolar	all other atoms

Table 4: Hydrogen bond types.

The PLP hydrogen bond term mentioned above only depends on the distance between atoms. In order to take into account the directionality of the hydrogen bonding, the geometry of the hydrogen bond is examined and the following factor H_{factor} is multiplied to the PLP hydrogen bond strength:

$$H_{factor} = \Phi(\angle_{D\text{-}H\text{-}A};90°;100°)\cdot\Phi\ (\angle_{H\text{-}A\text{-}AA};90°;100°)\cdot\ \Phi\ (\angle_{D\text{-}A\text{-}AA};90°;150°)$$

Here *AA* (Acceptor Antecedent) denotes a heavy atom connected to the acceptor (A), *D* denotes the donor and *H* is the donated hydrogen atom. The ramp function Φ is defined as $\Phi(A;A_{min};A_{max}) = 0$ for $A \le A_{min}$ and $\Phi(A;A_{min};A_{max}) = 1$ for $A \ge A_{max}$ and is linearly interpolated between these values for $A_{min} < A < A_{max}$. If it is not possible to calculate one of these factors it is omitted. This is for example the case for hydroxyl rotors where the exact location of the hydrogen is not investigated during docking, and the two first factors cannot be calculated. The angle checks above were motivated by the approach taken by McDonald and Thornton [MCDONALD 1994].

E_{intra} is the internal energy of the ligand:

$$E_{intra} = \sum_{i\in ligand}\ \sum_{j\in ligand} E_{PLP}(r_{ij}) \ + \sum_{flexible\ bonds} A[1-\cos(m\cdot\theta-\theta_0)] \ + \ E_{clash}$$

The double summation is between all atom pairs in the ligand excluding atom pairs which are connected by two bonds or less. The second term is a torsional energy term, parameterized according to the hybridization types of the bonded atoms (see Table 5). θ is the torsional angle of the bond. Notice that this angle is not necessarily uniquely determined. The average of the torsional energy bond contribution was used if several torsions could be determined. The last term, E_{clash}, assigns a penalty of 1000 if the distance between two heavy atoms

(more than two bonds apart) is less than 2.0 Å. Thus, E_{clash} term punishes infeasible ligand conformations.

	θ_0	m	A
sp^2-sp^3	0.0	6	1.5
sp^3-sp^3	π	3	3.0

Table 5: Torsional parameters.

12 Appendix II: Docking Search Algorithm

The docking search algorithm (MolDock Optimizer) used in MVD is based on an *evolutionary algorithm* [MICHALEWICZ 1992,2000].

Evolutionary algorithms (EAs) are iterative optimization techniques inspired by Darwinian evolution theory. In EAs, the evolutionary process is simplified and thus it has very little in common with real world evolution. Nevertheless, during the last fifty years EAs have proved their worth as powerful optimization techniques that can assist or replace traditional techniques when these fail or are inadequate for the task to be solved.

Basically, an EA consists of a population of individuals (candidate solutions), which is exposed to random variation by means of variation operators, such as mutation and recombination. The individual being altered is often referred to as the *parent* and the resulting solution after modification is called the *offspring*. Sometimes more than one parent is used to create the offspring by recombination of solutions, which is also referred to as *crossover*. Figure 65 below shows an outline of the *evolutionary process* taking place in EAs.

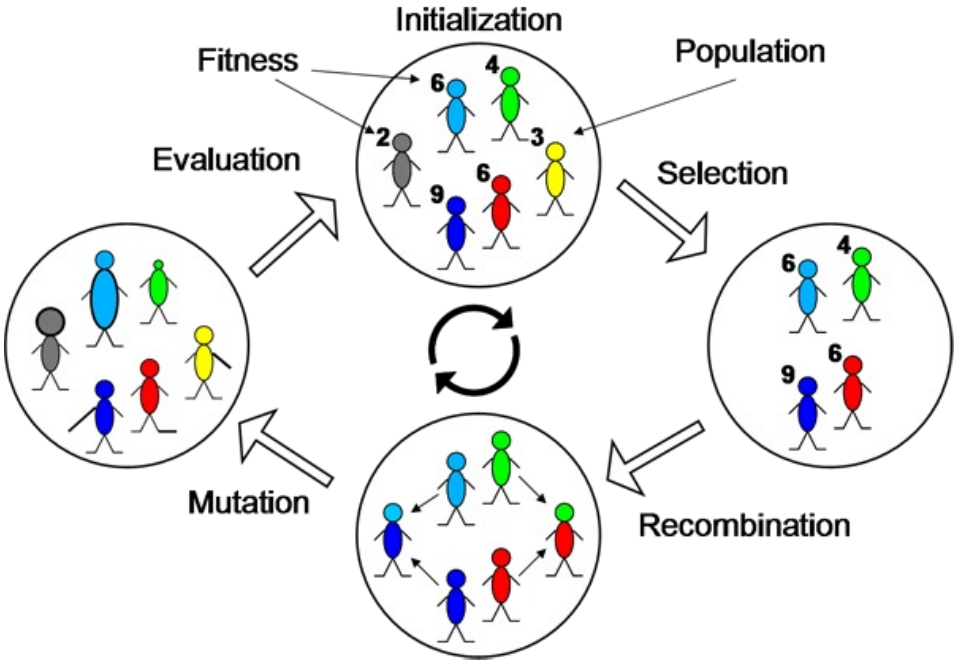


Figure 65: Outline of evolutionary algorithm.

Guided Differential Evolution

The guided differential evolution algorithm (MolDock Optimizer) used in MVD is based on an EA variant called differential evolution (DE). The DE algorithm was introduced by Storn and Price in 1995 [STORN 1995]. Compared to more widely known EA-based techniques (e.g. genetic algorithms, evolutionary programming, and evolution strategies), DE uses a different approach to select and modify candidate solutions (individuals). The main innovative idea in DE is to create offspring from a weighted difference of parent solutions.

The DE works as follows: First, all individuals are initialized and evaluated according to the MolDock Score (fitness function). Afterwards, the following process will be executed as long as the termination condition is not fulfilled: For each individual in the population, an offspring is created by adding a weighted difference of the parent solutions, which are randomly selected from the population. Afterwards, the offspring replaces the parent, if and only if it is more fit. Otherwise, the parent survives and is passed on to the next generation (iteration of the algorithm).

Additionally, guided differential evolution may use a cavity prediction algorithm (introduced in Appendix III) to constrain predicted conformations (poses) during the search process. More specifically, if a candidate solution is positioned outside the cavity, it is translated so that a randomly chosen ligand atom will be located within the region spanned by the cavity. Naturally, this strategy is only applied if a cavity has been found. If no cavities are reported, the search procedure does not constrain the candidate solutions.

One of the reasons why DE works so well is that the variation operator exploits

the population diversity in the following manner: Initially, when the candidate solutions in the population are randomly generated the diversity is large. Thus, when offspring are created the differences between parental solutions are big, resulting in large step sizes being used. As the algorithm converges to better solutions, the population diversity is lowered, and the step sizes used to create offspring are lowered correspondingly. Therefore, by using the differences between other individuals in the population, DE automatically adapts the step sizes used to create offspring as the search process converges toward good solutions.

Representation

Only ligand properties are represented in the individuals since the protein remains rigid during the docking simulation. Thus, a candidate solution is encoded by an array of real-valued numbers representing ligand position, orientation, and conformation as Cartesian coordinates for the ligand translation, four variables specifying the ligand orientation (encoded as a rotation vector and a rotation angle), and one angle for each flexible torsion angle in the ligand (if any).

Initialization

Each individual in the initial population is assigned a random position within the search space region (defined by the user).

Initializing the orientation is more complicated: By just choosing uniform random numbers for the orientation axis (between -1.0 and 1.0 followed by normalization of the values to form a unit vector) and the angle of rotation (between -180° and +180°), the initial population would be biased towards the identity orientation (i.e. no rotation). To avoid this bias, the algorithm by Shoemake et al. [SHOEMAKE 1992] for generating uniform random quaternions is used and the random quaternions are then converted to their rotation axis/rotation angle representation.

The flexible torsion angles (if any) are assigned a random angle between -180° and +180°.

Algorithmic Settings

In MVD, the following default parameters are used for the guided differential evolution algorithm: *population size = 50*, *crossover rate = 0.9*, and *scaling factor = 0.5*. These settings have been found by trial and error, and are generally found to give the best results across a test set of 77 complexes.

13 Appendix III: Cavity Prediction

In order to determine the potential binding sites, a grid-based cavity prediction algorithm has been developed. The cavity prediction algorithm works as follows:

First, a discrete grid with a resolution of 0.8 Å covering the protein is created. At every grid point a sphere of radius 1.4 Å is placed. It is checked whether this sphere will overlap with any of the spheres determined by the Van der Waals radii of the protein atoms. Grid points where the probe clashes with the protein atom spheres will be referred to as part of the inaccessible volume, all other points are referred to as accessible.

Second, each accessible grid point is checked for whether it is part of a cavity or not using the following procedure: From the current grid point a random direction is chosen, and this direction (and the opposite direction) is followed until the grid boundaries are hit, checking if an inaccessible grid point is hit on the way. This is repeated a number of times, and if the percentage of lines hitting an inaccessible volume is larger than a given threshold, the point is marked as being part of a cavity. By default 16 different directions are tested, and a grid point is assumed part of a cavity if 12 or more of these lines hit an inaccessible volume. The threshold can be tuned according to how enclosed the found cavities should be. A value of 0% would only be possible far from the protein as opposed to a value of 100% corresponding to a binding site buried deeply in the protein.

The final step is to determine the connected regions. Two grid points are connected if they are neighbours. Regions with a volume below 10.0 $Å^3$ are discarded as irrelevant (the volume of a connected set of grid points is estimated as the number of grid point times the volume of a unit grid cell). The cavities found are then ranked according to their volume.

14 Appendix IV: Clustering Algorithm

The multiple poses returned from a docking run are identified using the following procedure:

- During the docking run, new candidate solutions (poses) scoring better than parental solutions (see Appendix II: Docking Search Algorithm for details) are added to a temporary pool of docking solutions.
- If the number of poses in the pool is higher than 300, a clustering algorithm is used to cluster all the solutions in the pool (see below). The clustering is performed on-line during the docking search and when the docking run terminates. Because of the limit of 300 poses, the clustering process is fast. The members of the pool are replaced by the new cluster representatives found (limited by the **Max number of poses returned** option).

The clustering procedure works as follows:

1. The pool of solutions is sorted according to energy scores (starting with the best-scoring pose).
2. The first member of the sorted pool of solutions is added to the first initial cluster and the member is assigned to be the cluster representative.
3. The remainder of the pool members are added to the most similar cluster available (using the common RMSD measure) if and only if the RMSD between the representative of the most similar cluster and the member is below a user-specified RMSD threshold. Otherwise, a new cluster is created and the member is assigned to be the cluster representative.
4. The clustering procedure is terminated when the total number of clusters created exceeds **Max number of poses returned** (user-defined

parameter) or when all members of the pool have been assigned to a cluster.

5. When the cluster procedure has terminated, the set of representatives (one from each cluster) is returned.

15 Appendix V: Supported File Formats

MVD accepts the following molecular structure formats:

- PDB (Protein Data Bank). Supported file extensions: *pdb/ent*.
- Mol2 (Sybyl Mol2 format). Supported file extensions: *mol2*.
- SDF (MDL format). Supported file extensions: *sdf/sd* (for multiple structures) and *mol/mdl* (for a single molecular structure).

Currently, the following information is ignored during import of molecular structures:

- Lone pairs and dummy atoms (all file formats).
- When alternative atoms are reported, only the first alternative is used. The remainder is ignored (all file formats). If one of the other alternatives should be used, change the order of occurrence in the the file before import.
- CONNECT records (PDB format).
- SUBSTRUCTURE records are ignored during import but created when structures are exported (Mol2 format).

Notice: Although extensive testing and validation of the import and export of these file formats have been conducted, parsing errors may occur. Compliance with the file format standards/protocols will reduce parsing problems significantly. The import/export routines used have been extended to handle deviations from the file format protocols, but parsing errors may still occur. Found parsing errors can be reported (contact Technical Support or send email to bugs@molegro.com).

Additionally, Molegro Virtual Docker uses its own MVDML file format. MVDML is a shorthand notation for *Molegro Virtual Docker Markup Language* and is an

XML-based file format. In general, MVDML can be used to store the following information:

- Molecular structures (atom coordinates, atom types, partial charges, bond orders, hybridization states, ...)
- Constraints (location, type, and constraint parameters)
- Search space (center and radius)
- State information (workspace properties, ...)

Notice: Currently, the MVDML file format does not include cavities. Also, purely graphical objects (e.g. labels, interactions, annotations, backbones, and surfaces) are not saved.

16 Appendix VI: Automatic Preparation

The principles behind automatic preparation in MVD are described below.

Aromaticity

- All rings (closed loops) are identified.
- These rings are 'weeded out', until a 'smallest subset' (capable of covering all ring bonds) remains.
- These rings are considered aromatic if:
  1) For 5-cycles: the mean torsion angle is less then 9.5°
  2) For 6-cycles: the mean torsion angle is less then 12°
- If the aromatic ring contains an atom which has out-of-plane bonds, it is degraded to be non-aromatic.

Notice that this is only a geometrical check for aromacity. It does not include more advanced checks such as Hückel's rule, and may fail on overlapping ring systems.

Assign Hybridization

- All atoms with average bond angles > 155°, are marked as SP1
- All atoms with average bond angles > 115°, are marked as SP2
- All remaining atoms are marked SP3.
- All atoms part of aromatic rings are marked as SP2.
- Ensure that if an atom is SP2 or SP, it must be connected to another SP or SP2 or a terminal atom. Otherwise the atom is degraded (i.e. SP2 -> SP3)

- Lastly the geometry surrounding a SP2 atom should be planar, otherwise it is degraded to SP3.

Bond Order

- All atom bonds are set to 'unknown'. All implicit hydrogens are set to '-1'.
- All bonds to SP3 atoms are set to 'single' order.
- Next, a template file containing standard chemical motifs (-POO-, C(NH2)(NH2), ...) is processed. The templates are located in the file: `\misc\data\preparationTemplates.xml`
- All unset SP2-SP2 bonds involved in a planar geometry (less than 10 degrees) are set to 'double'.
- Next all SP2 atoms are checked to see if a double bond to a neighbour atom is possible. If several atom bonds are possible, the atom with highest electro negativity is chosen. If this still results in several possibilities, the atom closest to the current one will be chosen.

17 Appendix VII: Third Party Copyrights

MD5

MVD uses a derivate of the MD5 hash algorithm "RSA Data Security, Inc. MD5 Message-Digest Algorithm", under the following license:

You may use this software free of any charge, but without any warranty or implied warranty, provided that you follow the terms of the original RSA copyright, listed below.

Original RSA Data Security, Inc. Copyright notice

Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function. License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work. RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind. These notices must be retained in any copies of any part of this documentation and/or software.

Icons

The icon set used in MVD 2006 is taken from:

The Tango Icon Library: http://tango-project.org

They are released under the 'Creative Commons Share-Alike license':
http://creativecommons.org/licenses/by-sa/2.5/

18 Appendix VIII: Keyboard Shortcuts

The following list contains the keyboard shortcuts available in MVD. On Mac OS X, the CTRL key is replaced by the command key.

- CTRL-O Import Molecules
- CTRL-SHIFT-O Open Workspace
- CTRL-SHIFT-C Clear Workspace
- CTRL-S Save Workspace
- CTRL-F Toggle full screen
- CTRL-H Toggle dockable windows
- CTRL-C Toggle Cofactors category on/off
- CTRL-L Toggle Ligands category on/off
- CTRL-P Toggle Proteins category on/off
- CTRL-W Toggle Water category on/off
- CTRL-Z Undo
- CTRL-Y Redo
- CTRL-Q Quit MVD
- CTRL-1 to 8 Invoke misc. visualization views
- F1 to F9 Invoke misc. dialogs

Notice: Some of the shortcuts can be modified from the **Macro and Menu Editor** and additional shortcuts can be defined for macro commands.

19 Appendix IX: Console and Macro Commands

When using the **Macro and Menu Editor**, or entering commands in the console, the following commands can be used.

Notice: Some commands require a *molecule target*: these can be described using the following syntax:

Ligand[0] – the ligand with ID 0.

Ligand[4,5,6] – the Ligands with IDs 4,5 and 6. Multiple IDs are separated by comma.

Ligands – All ligands. By using the plural form of a category, all molecules in it are selected. The categories are: Pose, Cofactor, Protein, Water, Ligand.

Poses;Cofactors;Proteins;Ligands;Water[0]– All Poses, Cofactors, Proteins, Ligands and the first Water molecule. Multiple targets can be concatenated using a semi-colon.

Notice: The IDs of molecules are based on the order of occurrence in the corresponding **Workspace Explorer** category. For instance, ligand molecules listed in the **Ligands** category, begins with index 0 with increments of 1 (i.e. 0,1,2,3,...). If molecules are removed from the workspace, the IDs of the molecules are changed to follow the new order of occurrence in the list.

Command	Description
SET [active|reference] [targetligand]	Set active or reference ligand. A lot of operations (e.g. some surfaces) are only performed on the active ligand. The reference ligand is used to calculate RMSDs while docking.
EXPORT [moleculetarget]	Export as Mol2 or PDB. A File export dialog is opened for selection of a filename.
IMPOSECONSTRAINTS [TargetPose]	Can be used to test the current visible constraint. It tries to enforce all hard constraints, and summarizes the soft constraint energy in the console.
SURFACEDIALOG	Shows the Surface dialog.
PREPAREDIALOG	Shows the Preparation wizard.
DISTANCECONSTRAINT	Shows the Distance constraint dialog.
LABELDIALOG	Shows the Label dialog.
DOCKINGWIZARD	Shows the Docking Wizard.
GETPDB <key>	Downloads PDB with 'key' (4 letter code) from the Protein Data Bank.
ALIGN [MoleculeTarget1] [id1][id2] [id3] [MoleculeTarget2] [id1][id2] [id3]	Aligns atom id1,id2,id3 in MoleculeTarget1 with atom id1,id2,id3 in MoleculeTarget2.
SHOW CATEGORY <category> HIDE CATEGORY <category>	Shows or hides Workspace Explorer category with given name: i.e. SHOW CATEGORY water
REMOVE OBJECT [id]	Removes a 3D object from the world.
REMOVE [moleculetarget]	Removes an object from the workspace.
TRANSLATE [x] [y] [z]	Moves active ligand.
ROTATE [x] [y] [z] [angle]	Rotates active ligand.

CD	Print current directory.
DIR	Shows dir of MVDML files in current directory.
PREV	Loads previous MVDML file in current directory.
NEXT	Loads next MVDML file in current directory.
VALIDATE	Performs a validation of the complex.
RMSD	Invokes RMSD dialog.
CAV	Invokes Cavity detection dialog.
EVALUATOR [initstring]	Sets initstring for evaluator.
OPTIMIZER [initstring]	Sets initstring for optimizer.
DUMP [ligand|pose| ligandenergy| poseenergy|torsiontree]	Shows various debug info about structure.
EVAL [all|hbond| esbonds] EVALLIG [all|hbond| esbonds]	EVAL evaluates the energy of the active ligand. If 'all' is specified hbonds and electrostatic interactions are visualized. If 'hbond' is specified hbonds are visualized. If 'esbonds' is specified electrostatic interactions are visualized. EVALLIG evaluates the current pose (can be used while docking).
SELECT ID <id> SELECT ATOM <x y z> SELECT RESIDUE <id> SELECT RESIDUEID <id>	Selection of objects: 'SELECT ID' selects all atoms with id = 'id'. 'SELECT ATOM' selects closest atom to specified x, y, z position. 'SELECT RESIDUE' selects residue with residue index = 'id'. 'SELECT RESIDUEID' selects residue with internal residue index = 'id'.
SEED [number]	Sets random seed. It shows the current random seed if called without arguments.
STATUS	Shows info about the objects in the workspace and Visualization Window. Loaded modules are also listed.

SAVE [filename]	Saves a MVDML file. Do not include extension in filename.
LOAD [filename]	Loads a MVDML file. Do not include extension in filename.
ADD BACKBONE {radius}	Adds a backbone (a bezier-spline through the backbone carbons).
ADD CHARGES	Shows spheres around charged atoms.
ADD {P}SURF{*}{+} {radius} {resolution} {probesize} ADD {P}SURF{*}{+} LIGAND {resolution} {probesize}	Adds a molecular surface. *Notice: It is much easier to use the Surface dialog in the GUI*. If prepended by 'p' the surface will be colored by electrostatic potential. If followed by '*' the surface will *carved* (a Connolly surface). We recommend the carved surfaces for best visualization. If not followed by '+' existing surfaces will be removed. If no radius is given, the surface will cover the protein. If a radius is given, the surface will cover the *protein* in the given radius, but centered at the current active ligand! If the variant with 'LIGAND' argument is used, the surface will cover the currently active ligand. Resolution is typically 0.4 - 0.9. Don't choose higher resolutions (i.e. lower) than 0.4 unless you are prepared to wait for a long time! The default value of the probesize (1.2) should be fine for most purposes. Examples: ADD PSURF*+ 10 (Electrostatic carved protein surface with radius 10Å centered at the active ligand) ADD SURF* ligand 0.4 (Carved ligand surface with resolution 0.4)
DELETE [Charges|HBOND|Labels|Poses]	Deletes specified objects.
LIST POSES	Shows all poses with info about parameters.
CLS	Clears console log.
CLEAR [workspace|selection]	'CLEAR workspace' removes all items in the current workspace. 'CLEAR selection' clears current selection.
HIDE [hydrogens | labels]	Hides either hydrogens or labels.

SHOW [hydrogens | labels]	Shows either hydrogens or labels.
FITTOSCREEN	Fit all molecules in the visualization window.
LABEL	Used for labeling objects. This command is described in detail in the paragraph below. *Notice: It is much easier to use the Label dialog in the GUI*.
GUI Commands	
SLAB [near] [far]	Creates a slab (slicing) of the 3D world. *Notice: The Clipping Planes dialog is easier to use.*
QUALITY [value]	Sets OpenGL rendering quality from 0 to 10.
LIGHT [number] [on|off] [ambient] [diffuse] [specular] {[x] [y] [z]}	Sets OpenGL light sources.
FOG LINEAR [near] [far] FOG [EXP|EXP2] [exponent] FOG OFF	Sets OpenGL fog.
COLOR [protein|pose| ligand|water|cofactor] [fixed|cpk|hbond| hbond2|interaction| interaction2] {r g b}	Sets the color style of specified object.
STYLE [protein|pose| ligand|water|cofactor] [vdw, fixed, stick, none] atomScale bondScale	Sets the visualization style of specified object.
REBUILD	Rebuilds all objects in the Visualizer Window. This command is necessary to call after the visualization styles or coloring schemes have been updated. Otherwise graphical changes will not be reflected in the GUI.

The label command works in the following way: it scans the input-string for known variables (like ID, HYB, ELE - see below) and replaces them with their value. That is, the command 'label bond bond_number:id' will add a label of type 'bond number x' to every bond (underscores are replaced with spaces).

To clear all labels use 'label' without any argument.

Variable	Description
Atom labels. Syntax: '**Label *string***'	
ID	Internal atom index
Type	Hydrogen bond type: non-polar, acceptor, donor, both. The HBOND variable below is probably of more use.
PC	Partial Charge.
PC!	PC! ignores atoms with no partial charge.
HYB	Hybridization.
HYB!	HYB! only displays hybridization for atoms with other hybridizations than SP3 or unknown.
SP2	Labels SP2 hybridized atoms
SYM	Element symbol. (H, C, N, ...)
ELE	Element number.
IH	Number of implicit hydrogens.
HBOND	Hydrogen bond type shown as : D, A, D+A, - (non-polar)
HBOND!	HBOND! ignores non-polar atoms.
ETOT	Shows the total energy of the atom. This requires that the energy has been evaluated using the 'eval' command.
Bond labels. Syntax: '**Label bond *string***'	
ID	Internal bond index.
Type	Bond order: single, double, triple, aromatic, ...
ETOT	Shows the total energy of the bond. This requires that the energy has been evaluated using the 'eval' command.
Torsion Tree Labels. Syntax: '**Label tree *string***'. Torsion Tree labels are evaluated at each rigid component of the ligand.	
EA, *EA	EA shows the Atom Energy of the rigid component . EA* shows the difference between this energy for the pose and for the

	ligand. *It is only well-defined on poses.*
EB, *EB	EB shows the Bond Energy of the rigid component. EB* shows the difference between this energy for the pose and for the ligand. *It is only well-defined on poses.*
ETOT, *ETOT	Shows the total energy of the rigid component of the tree. This requires that the energy has been evaluated using the 'eval' command. ETOT* shows the difference between this energy for the pose and for the ligand. *It is only well-defined on poses.*
DEPTH	The depth of the torsion tree. Higher values indicate that the rigid component is built later. The rigid root component will have depth 0. Optimal placement of the rigid root will result in a lower maximum depth.
BOND	Shows the bond index that this rigid component rotates about.
ID	Internal index
ATOMS	Number of atoms in rigid component.
Residue Labels. Syntax: '**Label residue *string***'	
ID	Internal residue index
LONGNAME	Full residue name ('histidine', 'cysteine', ...)
NAME	3-letter abbreviation ('HIS', 'CYS', ...)
LETTER	1-letter abbreviation.

20 Appendix X: Script Commands

This appendix describes all the script commands that are available in MVD.

Some script commands require a *molecule target*: these can be described using the following syntax:

Ligand[0] – the ligand with ID 0.

Ligand[4,5,6] – the Ligands with IDs 4,5 and 6. Multiple IDs are separated by comma.

Ligand[50-60] – the Ligands with IDs from 50 to 60 (both included). Ligand ranges are specified by a "-".

Ligands – All ligands. By using the plural form of a category, all molecules in it are selected. The categories are: Pose, Cofactor, Protein, Water, Ligand.

Poses;Cofactors;Proteins;Ligands;Waters – All Poses, Cofactors, Proteins, Ligands and all Water molecules. Multiple targets can be concatenated using a semi-colon.

All – imports all structures.

Notice: The IDs of molecules are defined by their order of occurrence in the workspace. All indices are zero-based, meaning that the first ligand will have index 0, the second index 1, and so forth.

20.1 List of Script Commands Available

Comments in MVD script files

It is possible to add comments to MVD script files using either // for a one line comment or /* */ to span more line.

Notice: Currently, it is not possible to add comments after script commands.

Examples:

```
// This is a one line comment
```

```
/* This is a comment spanning more than one line which can be useful
when describing what is going on */
```

IMPORT <targets> FROM <file>

The IMPORT command reads molecular data from either PDB, ENT, MOL2, MOL, SDF, SD files.

<targets> is the usual syntax for specifying the molecules to import.

Notice:

- Files imported using the IMPORT command are always prepared using currently set preparation-settings. (See the PREPARE command).
- Files are always appended to the workspace. (The workspace is not cleared). You can clear the workspace using the NEW command.
- If you want to import MVDML files, use the LOAD command.
- If a complete file path is not specified, the current working directory is used to search for the files (See the CD command).

Examples:

```
IMPORT Ligand[1,3] FROM testdock.mol2
```

```
IMPORT All FROM testdock.mol2
```

```
IMPORT Proteins;Waters;Cofactors FROM 1hvr.pdb
```

PREPARE <settings-string>

Determines how molecules imported using the IMPORT command are prepared.

The settings-string is composed of semi-colon separated pairs of a preparation-type and its preparation scheme:

Preparation Types: **Bonds**, **BondOrders**, **Hydrogens**, **Charges**, or **TorsionTrees**

Preparation Schemes: **IfMissing**, **Always**, **Never**, or **Remove**

The default setting is:

```
PREPARE Bonds=IfMissing;BondOrders=IfMissing;Hydrogens=IfMissing;Charges=
Always; TorsionTrees=Always
```

It is not necessary to specify all of the PREPARE settings. If only some of them are specified the default parameters will be used for the remainder.

Examples:

```
PREPARE Bonds=Always
// Ensures that we use the built-in algorithm to determine atom connectivity.
```

```
PREPARE Charges=IfMissing
// Uses the charges from the molecular input file (default is to overwrite
them).
```

LOAD <mvdml-filename>

Loads a workspace from a MVDML file.

Notice that this command will replace the current workspace.

No preparation is done on the workspace, since it is assumed that files saved in MVDML format are prepared already.

Notice: LOAD clears the current workspace (if any).

SAVE <mvdml-filename>

Save the current workspace as a MVDML file.

All molecular structures in the workspace are saved.

EXIT

Causes the MVD process to terminate.

This can be useful if running several docking simulations of different proteins automated from a scripting language (i.e. using the Python wrapper.)

Do not use this command when parsing a text-file script as it will terminate the script and not parse anything after the EXIT command.

DOCK <molecules>

The DOCK command initiates the docking process. <molecules> is a list of ligands (notice only ligands are allowed here) to be docked. <molecules> is specified in the usual target format.

The settings for the docking can be modified using the DOCKSETTINGS command. The MolDock evaluator and optimizer can be modified using the EVALUATOR and OPTIMIZER commands.

Examples:

```
DOCK Ligand[50-60]
// Docks ligand from number 50 to number 60 (both included) in the current workspace
```

```
DOCK Ligand[0]
// Docks first ligand in the current workspace
```

```
DOCK Ligands
// Docks ALL ligands in workspace
```

CD <path>

Changes the current working directory to the given path.

EVALUATOR <initstring>

Sets the settings for the evaluator (the docking score function).

There is normally no need to change these.

The **<initstring>** is semi-colon separated string of parameter-value pairs.

The following parameters are available. Their default setting is marked in bold face:

ligandes = [**true**|false]. Determines whether the internal electro static energy of the ligand should be included

torsion = first, **mean**, all. Determines how torsion terms are evaluated (if several torsion angles are available for a bond).

sp2sp2bond = [**true**|false]. Determines if sp2-sp2 bonds should be taken into account.

eintra = [**true**|false]. Determines whether ligand self-interaction energy should be taken into account.

cropdistance [double]. Determines whether the protein should be cropped (meaning protein atoms outside a given distance is not taken into account). If crop distance is 0 (the default settings) the size of the active search space is used. Other the crop distance is defined from the center of the current reference ligand. Crop distance is measured in Ångstrom. If crop distance is negative, all atoms in the protein will be taken into account. Notice that the docking duration increases with the number of atoms. It is advised to keep the default settings of 0.

skiptorsionterm = [true|**false**]. Determines whether ligand torsions are taken into account.

hbond90 = [**true**|false]. Determines whether hydrogen bonding directionality should be taken into account. *Notice:* The **hbond90** option is not available for the grid evaluator.

water = [**true**|false]. Determines whether water should be taken into account.

In addition to these parameters, the following parameter option is available when using the Grid-based scoring function.

gridresolution = [double]. Sets the grid spacing, where the grid resolution is specified in Ångstrom.

EVALUATOR <initstring>

The default settings from the docking wizard will generate the following evaluator string:

```
EVALUATOR cropdistance=0;hbond90=true;water=true
```

Notice: an easy way to generate a suitable initstring is to use the Docking Wizard to generate and save a generated script.

EVALUATORTYPE <type>

The EVALUATORTYPE command set the evaluator (scoring function) used while docking.

<type> is one of the following values:

- **MolDockGrid** – for the grid version of the evaluator.
- **MolDock** – for the standard version.

Notice: MolDock is set automatically as the default evaluator.

Example:

```
EVALUATORTYPE MolDockGrid
```

OPTIMIZER <initstring>

Sets the settings for the optimizer (the docking search algorithm).

The **<initstring>** is semi-colon separated string of parameter-value pairs.

The following parameters are available. Their default setting is marked in bold. For more information about the parameters see Appendix II: Docking Search Algorithm, Appendix XI: MolDock SE, or the section about the Docking Wizard where the parameters also are described.

popsize [integer=**50**]. Determines the number of individuals in the population.

cavity [**true**|false]. Determines whether poses should be forced to be in cavities.

scalingfactor [double=**0.50**].

crossoverrate [double=**0.90**].

offspringstrategy [int=**1**]

earlytermination [double=**0.01**]

terminationscheme [int=**0**]

randomizeligand [**true**|false]. Determines whether the ligand orientation should be randomized before each docking run.

keepmaxposes [int=**5**]

excludeenergythreshold [double=**10000**]

clusterthreshold [double= **0.0**]

The following parameters are used by the MolDock SE search algorithm:

creationenergythreshold=[double]. Default is **100.0**. Poses are only added to the population if the value is this threshold. Notice that when half of the iterations in the docking run have been used, this threshold is automatically turned off in order to ensure that enough poses are created for the simplex evolution phase.

posegenerator=[int,int,int]. Set the Min, Quick, Max number of tries. Default is **10,10,30**. At each step at least 'min' torsions/translations/rotations are tested and the one giving lowest energy is chosen. If the energy is positive (i.e. because of a clash or unfavorable electrostatic interaction) then additional 'max' positions will be tested. If at one time it has not been possible to construct a component which do not clash, the 'max' tries number is lowered to the 'quick' try value.

simplexsteps=[int (default:300)]. The number of iterations of the Nelder-

OPTIMIZER <initstring>

Mead simplex minimization procedure performed at each step of the MolDock SE algorithm.

simplexdistancefactor=[double]. Default is **1.0**. This factor determines how close the point of the initial simplex will be to the other randomly selected individuals in the population. A factor of 1.0 causes the initial simplex to span the neighbour points exactly, while a factor of 0.5 would correspond to simplex points being created halfway between the individuals chosen for optimization and its randomly chosen neighbours. Notice that a factor less than 1.0 will converge slowly. Typical values should be in the range of 0.95 to 3.0.

recombine=[**true** | false]. Allows for turning off the Simplex Evolution phase.

The default settings (using the MolDock search algorithm) from the Docking Wizard will generate the following optimizer string:

```
OPTIMIZER cavity=false;popsize=50;scalingfactor=0.50;crossoverrate=0.90;
offspringstrategy=1;terminationscheme=0;earlytermination=0.01;
clusterthreshold=1.00;keepmaxposes=5
```

Another example using the MolDock SE search algorithm:

```
OPTIMIZER
populationsize=50;cavity=true;creationenergythreshold=100;posegenerator=10,10
,30;maxsimplex=750;simplexsteps=300;simplexdistancefactor=1
```

Notice: an easy way to generate a suitable initstring is to use the Docking Wizard to generate and save a generated script.

NEW

Clears the current workspace:

All molecules are removed from the workspace.

DOCKSETTTINGS <initstring>

Determines the behavior of the docking engine.

The settings-string is composed of semi-colon separated pairs of a parameter key and its corresponding value. The different parameters are:

maxIterations: the value must be an integer describing the maximum number of iterations by the MolDock engine. The default value is 2000.

runs: the number of runs performed for each ligand. Multiple runs will giver higher docking accuracy. The default number is 1. Typically 5 to 10 runs are recommended.

ignoreSimilarPoses: when running multiple runs, several poses are returned for each ligand. Set this to 'true' to weed out similar poses by clustering according to their RMS deviation. Default value is 'true'

IgnoreSimilarPosesThreshold: This is the RMSD treshold value in Ångstrom for the clustering described above. Default value is 1.

MaxPoses: Determines the maximum number of poses returned by the clustering described above. Default value is 5

The default settings corresponds to the following script command:

```
DOCKSETTINGS maxIterations=2000;runs=1;ignoreSimilarPoses=true;
IgnoreSimilarPosesThreshold=1.0;MaxPoses=5
```

It is not necessary to specify all of the parameters. If only some of them are specified the default parameters will be used for the remainder.

Examples:

```
PREPARE maxIterations=4000
// Use a higher number of iterations
```

```
PREPARE runs=10
// Multiple runs increases the accuracy of the poses found.
```

OPTIMIZERTYPE <type>

The OPTIMIZERTYPE command sets the optimizer (search function) used while docking.

<type> is one of the following values:

- **MSE** – for the MolDock SE algorithm.
- **MolDock** – for the standard MolDock algorithm.

Notice: MolDock is set automatically as the default optimizer.

Example:

```
OPTIMIZERTYPE MSE
```

RANDOM <seed>

Sets the seed used by the random number generator.

Normally this is not recommended, since a random seed always is generated on startup, but it can be used to reproduce docking runs, since the seed is always recorded in the docking log.

```
RANDOM 123
// Ensures that the simulation will always return the exact same results.
```

INFO <output>

Writes "output" to the console.

Can be useful for debugging loops.

Example:

```
INFO Variable a is $a
// Outputs the value of "$a"
```

CONSTRAINTS <integer list>

Per default all constraints defined in a MVD workspace are used.

The CONSTRAINTS command enables a subset of the constraints in the workspace. All constraints not specified in the list are not used during the docking run.

To disable all constraints, set “integer list” = “NONE”.

It is possible to specify ranges, or to just enable all constraints by setting “integer list” = “ALL”.

Notice: the numbering of constraints is zero-based, meaning that the first constraint in a workspace will have number 0, the second number 1 and so forth.

Examples:

```
CONSTRAINTS 1,2
/* Enables the second and third constraint in the workspace
   All other constraints are disabled */
```

```
CONSTRAINTS 1,3-5
/* Enables the second, fourth, fifth and sixth constraint in the workspace
   All other constraints are disabled */
```

```
CONSTRAINTS NONE
// Disables all constraints in the workspace
```

```
CONSTRAINTS ALL
// Enables all constraints in the workspace (default behavior)
```

RMSD <targetligand>

The RMSD can be used to set a ligand to compare docking results with.

The Root-Mean-Square-Deviation between heavy atoms will be calculated for all returned poses.

Notice: the ligand used as reference for RMSD calculations must have the same number of heavy atoms as the ligands that are docked, otherwise the RMSD calculation will just return -1.

Examples:

```
LOAD 3PTB.MVDML
RMSD ligand[1]
DOCK
// Docks the ligands in 3PTB.MVDML and calculate their RMSD deviation from
ligand[1], which is the second ligand present in the workspace
```

DOWNLOAD <PDB code> AS <filename.pdb>

The DOWNLOAD command can be used to download a PDB file from the Protein Data Bank. The downloaded file will be saved as <filename.pdb>.

The downloaded PDB file is not automatically imported to the current workspace. This should be done using the IMPORT command.

Notice: the <PDB code> is a 4-letter PDB identifier and that the filename should include the pdb file extension.

Moreover, the DOWNLOAD command overwrites existing filenames named <filename.pdb>.

Examples:

```
DOWNLOAD 3ptb AS 3ptb.pdb
IMPORT All FROM 3ptb.pdb
DOCK
```

20.2 Flow Control

MVD also provides a couple of simple commands for controlling the script flow. If more complex execution control is needed consider using the Python wrapper to control to scripting engine.

Notice: The variable system in the script parser is strictly string based which means that the script parser simply substitutes occurrences of variable names with the current value before parsing the string.

Also notice that this means that it is important to be careful when defining variable names and ensure that they do not overlap: e.g. do not define two variables named $PDB and $PDBS since the script parser will substitute part of the variable name $PDBS with the value of $PDB.

FOR <$VAR> IN <VALUELIST>
ENDFOR

The FOR command can be used to iterate though a set of possible values.

The VALUELIST must be a comma separated list of values.

FOR commands can be nested (it is possible to have a FOR command inside another FOR loop).

Variables must start with a "$" identifier.

Example (docking multiple complexes):

```
FOR $PDB IN 3PTB,1HVR,1LIC,1TMN
   // $PDB will be replaced by the appropriate value in the loop
   LOAD C:\BENCHMARK\$PDB.mvdml
   RMSD ligand[0]
   DOCK
   NEW
ENDFOR
```

Example (docking with different population sizes):

```
FOR $popsize IN 10,20,30,40,50
   OPTIMIZER cavity=true;popsize=$popsize;crossoverrate=0.9;
   LOAD C:\BENCHMARK\3PTB.mvdml
   RMSD ligand[0]
   DOCK
   NEW
ENDFOR
```

SET <$VAR> = <VALUE>

The SET command can be used to set a variable to a given value.

Variables must start a "$" identifier.

Example:

```
SET $PDB = 3PTB
LOAD C:\BENCHMARK\$PDB.mvdml
RMSD ligand[0]
DOCK
```

21 Appendix XI: MolDock SE

MolDock SE (simplex evolution) is an alternative search heuristic which can be used together with either the *MolDock* or *MolDock [Grid]* scoring functions.

It is known to perform better on some complexes where the standard MolDock algorithm fails. This is usually the case when the ligand has lots of internal degrees of freedom (many torsion angles).

Notice: MolDock SE is still experimental, as all parameters have not been fine-tuned yet. Although it should be stable, it is still to be considered a technology preview version.

While other algorithms based on parallel simplex search exist, our implementation has been modified to be suitable for docking (by the inclusion of the pose generation step, and the way the initial simplices are created).

The algorithm works as follows:

Pose Generation

First an initial population of poses is created. The initial number of poses is determined by the 'population size' parameter.

These poses are built incrementally from their rigid root point: The pose generator tests a number of different torsions angles, rotations and translations, evaluates the affected part of the molecule and chooses the value which results in the lowest energy contribution.

The torsion angles are chosen from one of three distributions depending on the hybridization of the atoms the bond connects (either sp2-sp2, sp2-sp3 or sp3-sp3).

If the generated pose has an energy below 'energy threshold' it is accepted

into the initial population for the 'simplex evolution' algorithm.

Simplex Evolution

The simplex evolution algorithm performs a combined local / global search on the poses generated by the pose generator. The local search is performed using the Nelder-Mead local search algorithm, but unlike Nelder-Mead's original scheme, the algorithm has been extended to take the position of the other individuals in the population into account. At each iteration a random individual is chosen. The representation of this individual determines the first point of the simplex in the N-dimensional search space. Then N additional individuals are chosen and their representations define the remaining N points of the simplex (a simplex in N dimensions has N+1 points). Notice that 'Neighbour distance factor' parameter determines how much the initial simplex should be enlarged or shrinked (see below).

Usage From GUI

In order to use the new search algorithm choose '**Search algorithm** -> **Algorithm** -> **MolDock SE (experimental)**' from the **Docking Wizard**.

The following parameters can be set:

Max iterations: (default=1500) The number of steps per run. These steps are evenly divided between the pose generator and the simplex evolution algorithm (even though both of these may terminate before the number of iterations has been used).

Max population size: (default=50) The number of individuals in the simplex evolution phase. Notice that this number must be higher than the number of degrees of freedom (7 spatial degrees of freedom plus the number of chosen rotatable torsion bonds).

Pose Generation Parameters

Energy threshold: (default=100.00) Poses are only added to the population if the value is below this threshold. Notice that when half of the iterations in the docking run have been used, this threshold is automatically turned off in order to ensure that enough poses are created for the simplex evolution phase.

Tries: **Min**, **Quick**, **Max**.

At each step at least 'min' torsions/translations/rotations are tested and the one giving lowest energy is chosen. If the energy is positive (i.e. because of a clash or an unfavorable electrostatic interaction) then additional 'max' positions will be tested. If it is not possible to construct a component which do not clash, the 'max' tries number is lowered to the 'quick' try value.

Simplex Evolution Parameters

Max Steps: (default=300). The number of iterations of the Nelder-Mead simplex minimization procedure performed at each step of the MolDock SE algorithm.

Neighbour distance factor: (default=1.0). This factor determines how close the point of the initial simplex will be to the other randomly selected individuals in the population. A factor of 1.0 causes the initial simplex to span the neighbour points exactly, while a factor of 0.5 would correspond to simplex points being created halfway between the individuals chosen for optimization and its randomly chosen neighbours. Notice that a factor less than 1.0 will converge slowly. Typical values should be in the range of 0.95 to 3.0.

Usage When Scripting

To use the MolDock SE search algorithm, the OPTIMIZERTYPE script command has to be set. Moreover, specific search algorithm parameters are set by the OPTIMIZER script command (see Appendix X: Script Commands for more details).

22 Appendix XII: MolDock Score [Grid]

'MolDock Score [Grid]' is a grid based version of the MolDock Score function.

It precalculates potential-energy values on an evenly spaced cubic grid in order to speed up calculations. Notice that unlike the standard MolDock Score, the grid version does not take hydrogen bond directionality into account (hydrogen bonding is determined solely on distance and hydrogen bonding capabilities). The energy potential is evaluated by using tri-linear interpolation between relevant grid points. The rest of the terms in the MolDock Score [Grid] version (i.e. internal ligand energy contributions and constraint penalties) are identical to the standard version of the scoring function.

Grids are not stored permanently - they are calculated when needed. (Grid generation is relatively fast. Typically ~15 seconds for the standard settings). Grids will automatically be reused while running docking scripts as long as the target protein does not change.

Notice that large energy grids with high resolution can consume a lot of memory. Grid resolutions of 0.3 Å -0.4 Å will be adequate in most cases. Look out for the estimated memory usage in the Docking Wizard. As a rule of thumb it should never exceed more than half of the physical memory available in the computer. Also notice that if several instances (processes) of MVD is running, each process will need to generate its own grid.

Usage From GUI

In order to use the MolDockScore grid version, select it as the evaluation function in the **Docking Wizard** ('**Scoring Function** -> **Score** -> **MolDock Score [GRID]**').

Usage When Scripting

To use the grid-based scoring function, the EVALUATORTYPE script command has to be set. Moreover, specific grid parameters are set by the EVALUATOR script command (see Appendix X: Script Commands for more details).

23 Appendix XIII: References

[THOMSEN 2006] Thomsen, R.; Christensen, M. H. MolDock: A New Technique for High-Accuracy Molecular Docking. *J. Med. Chem.*, 2006, 49(11), 3315-3321.

[CCG] Chemical Computing Group, www.chemcomp.com

[SCHRODINGER] Schrödinger, LLC, www.schrodinger.com

[GEHLHAAR 1995] Gehlhaar, D. K.; Verkhivker, G.; Rejto, P. A.; Fogel, D. B.; Fogel, L. J.; Freer, S. T. Docking Conformationally Flexible Small Molecules Into a Protein Binding Site Through Evolutionary Programming. *Proceedings of the Fourth International Conference on Evolutionary Programming* 1995, 615-627.

[GEHLHAAR 1998] Gehlhaar, D. K.; Bouzida, D.; Rejto, P. A. Fully Automated And Rapid Flexible Docking of Inhibitors Covalently Bound to Serine Proteases. *Proceedings of the Seventh International Conference on Evolutionary Programming* 1998, 449-461.

[YANG 2004] Yang, J-M.; Chen, C-C. GEMDOCK: A Generic Evolutionary Method for Molecular Docking. *Proteins* 2004, *55*, 288-304.

[MCDONALD 1994] McDonald, I. K.; Thornton, J. M. Satisfying Hydrogen Bonding Potential in Proteins. *J. Mol. Biol.* 1994, *238*, 777-793.

[MICHALEWICZ 1992] Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*; Springer-Verlag: Berlin, 1992.

[MICHALEWICZ 2000] Michalewicz, Z.; Fogel, D. B. *How to Solve It: Modern Heuristics*; Springer-Verlag: Berlin, 2000.

[STORN 1995] Storn, R.; Price, K. Differential Evolution - A Simple And Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Tech-report, International Computer Science Institute, Berkley, 1995.

[SHOEMAKE 1992] Shoemake, K. Uniform Random Rotations. In *Graphics Gems III*, 1st ed.; Kirk, D., Ed.; AP Professional (Academic Press); Boston, 1992; pp. 124-132.